Operating System

Active Directory Architecture

White Paper

Abstract

To use the Microsoft® Windows® 2000 Server operating system with maximum effectiveness, you must first understand what the Active Directory™ directory service is. Active Directory, new in the Windows 2000 operating system, plays a major role in implementing your organization's network and therefore in accomplishing its business goals. This paper introduces network administrators to Active Directory, explains its architecture, and describes how it interoperates with applications and other directory services.

© 2000 Microsoft Corporation. All rights reserved.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

Microsoft, Active Directory, ActiveX, BackOffice, MSN, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Other product and company names mentioned herein may be the trademarks of their respective owners.

Microsoft Corporation • One Microsoft Way • Redmond, WA 98052-6399 • USA

0x99

CONTENTS

INTRODUCTION....1
Active Directory Benefits 1

ACTIVE DIRECTORY DIRECTORY SERVICE....3
Active Directory Incorporates DNS 3
Active Directory Creates Domain Controller 8

ARCHITECTURE....13
Objects 13
Object Naming Conventions 16
Object Publishing 21
Domains: Trees, Forests, Trusts, and OUs 23
Sites: Service Clients and Replicate Data 32
Use Delegation and Group Policy with OUs, Domains, and Sites 40

INTEROPERABILITY....45
Lightweight Directory Access Protocol 45
Application Programming Interfaces 45
Synchronizing Active Directory with Other Directory Services 46
Internal and External References 47
Kerberos Role in Interoperability 48
Backward Compatibility with the Windows NT Operating System 48

SUMMARY....50
For More Information 50

APPENDIX A: TOOLS....52
Microsoft Management Console 52
Active Directory Snap-ins 52
New Ways to Do Familiar Tasks 53
Active Directory Command-line Tools 54
Windows 2000 Command Reference Page 55
Active Directory Service Interface 55

INTRODUCTION

Gaining an understanding of the Active Directory™ directory service is the first step in understanding how the Windows® 2000 operating system functions and what it can do to help you meet your enterprise goals. This paper looks at Active Directory from the following three perspectives:

- **Store.** Active Directory, the Windows 2000 Server directory service, hierarchically stores information about network objects and makes this information available to administrators, users, and applications. The first section of this paper explains what a directory service is, the integration of Active Directory service with the Internet's Domain Name System (DNS), and how Active Directory is actualized when you designate a server as a domain controller[1].
- **Structure.** Using Active Directory, the network and its objects are organized by constructs such as domains, trees, forests, trust relationships, organizational units (OUs), and sites. The next section in this paper describes the structure and function of these Active Directory components, and how this architecture lets administrators manage the network so that users can accomplish business objectives.
- **Inter-communicate.** Because Active Directory is based on standard directory access protocols, it can interoperate with other directory services and can be accessed by third-party applications that follow these protocols. The final section describes how Active Directory can communicate with a wide variety of other technologies.

Active Directory Benefits

The introduction of Active Directory in the Windows 2000 operating system provides the following benefits:

- **Integration with DNS.** Active Directory uses the Domain Name System (DNS). DNS is an Internet standard service that translates human-readable computer names (such as mycomputer.microsoft.com) to computer-readable numeric Internet Protocol (IP) addresses (four numbers separated by periods). This lets processes running on computers in TCP/IP networks identify and connect to one another.
- **Flexible querying.** Users and administrators can use the **Search** command on the **Start** menu, the **My Network Places** icon on the desktop, or the Active Directory Users and Computers snap-in to

[1] In a Windows 2000 Server domain, a domain controller is a computer running the Windows 2000 Server operating system that manages user access to a network, which includes logging on, authentication, and access to the directory and shared resources.

quickly find an object on the network using object properties. For example, you can find a user by first name, last name, e-mail name, office location, or other properties of that person's user account. Finding information is optimized by use of the global catalog.

- **Extensibility.** Active Directory is extensible, which means that administrators can add new classes of objects to the schema and can add new attributes to existing classes of objects. The schema contains a definition of each object class, and each object class's attributes, that can be stored in the directory. For example, you could add a Purchase Authority attribute to the User object and then store each user's purchase authority limit as part of the user's account.
- **Policy-based administration.** Group Policies are configuration settings applied to computers or users as they are initialized. All Group Policy settings are contained in Group Policy Objects (GPOs) applied to Active Directory sites, domains, or organizational units. GPO settings determine access to directory objects and domain resources, what domain resources (such as applications) are available to users, and how these domain resources are configured for use.
- **Scalability.** Active Directory includes one or more domains, each with one or more domain controllers, enabling you to scale the directory to meet any network requirements. Multiple domains can be combined into a domain tree and multiple domain trees can be combined into a forest. In the simplest structure, a single-domain network is simultaneously a single tree and a single forest.
- **Information Replication.** Active Directory uses multimaster replication, which lets you update the directory at any domain controller. Deploying multiple domain controllers in one domain provides fault tolerance and load balancing. If one domain controller within a domain slows, stops, or fails, other domain controllers within the same domain can provide necessary directory access, since they contain the same directory data.
- **Information security.** Management of user authentication and access control, both fully integrated with Active Directory, are key security features in the Windows 2000 operating system. Active Directory centralizes authentication. Access control can be defined not only on each object in the directory, but also on each property of each object. In addition, Active Directory provides both the store and the scope of application for security policies. (For more about Active Directory logon authentication and access control, see the "For More Information" section at the end of this paper.)
- **Interoperability.** Because Active Directory is based on standard

directory access protocols, such as Lightweight Directory Access Protocol (LDAP), it can interoperate with other directory services employing these protocols. Several application programming interfaces (APIs) —such as Active Directory Service Interfaces (ADSI) —give developers access to these protocols.

At the end of this document, “Appendix A: Tools” provides a brief overview of the software tools you use to perform the tasks associated with Active Directory.

Before getting to the main sections of this paper—Active Directory architecture and interoperability—this preliminary section takes a quick look at Active Directory from two very different perspectives:

- The first is Active Directory at its most abstract, that is, Active Directory as a namespace that is integrated with the Internet’s Domain Name System (DNS).
- The second is Active Directory at its most mundane, that is, as the software that makes a server into a domain controller.

In the context of a computer network, a *directory* (also called a data store) is a hierarchical structure that stores information about *objects* on the network. Objects include shared resources such as servers, shared volumes, and printers; network user and computer accounts; as well as domains, applications, services, security policies, and just about everything else in your network. One example of the specific kinds of information a network directory might store about a particular type of object is that a directory typically stores a user’s name, password, e-mail address, phone number, and so on, for a user account.

A *directory service* differs from a directory in that it is both the directory information source and the services making the information available and usable to administrators, users, network services, and applications. Ideally, a directory service makes the physical network topology and protocols (formats for transmitting data between two devices) transparent so that a user can access any resource without knowing where or how it is physically connected. To continue the user account example, it is the directory service that lets other authorized users on the same network access stored directory information (such as an e-mail address) about the user account object.

Directory services can support a wide variety of capabilities. Some directory services are integrated with an operating system, and others are applications such as e-mail directories. Operating system directory services, such as Active Directory, provide user, computers, and shared resource management. Directory services that handle e-mail, such as

Microsoft Exchange, enable users to look up other users and send e-mail.

Active Directory, the new directory service central to the Windows 2000 Server operating system, runs only on domain controllers. Active Directory, in addition to providing a place to store data and services to make that data available, also protects network objects from unauthorized access and replicates objects across a network so that data is not lost if one domain controller fails.

Active Directory Incorporates DNS

Active Directory and DNS are both namespaces. A *namespace* is any bounded area in which a given name can be resolved. Name resolution is the process of translating a name into some object or information that the name represents. A telephone book forms a namespace in which the names of telephone subscribers can be resolved to telephone numbers. The Windows 2000 NTFS file system forms a namespace in which the name of a file can be resolved to the file itself.

DNS and the Internet

Understanding how Windows 2000 handles Active Directory and DNS namespaces requires understanding a few basics about DNS itself and its relationship to the Internet and TCP/IP. The Internet is a TCP/IP network. The TCP/IP communications protocols connect computers and let them transmit data over networks. Every computer on the Internet or on any other TCP/IP network (such as many Windows networks) has an IP address. DNS locates TCP/IP *hosts* (computers) by resolving the computer names that end users understand to the IP addresses that computers understand. The IP addresses on the Internet are managed by using the globally distributed DNS database, but DNS can also be implemented locally to manage addresses within private TCP/IP networks.

DNS, which is organized into a hierarchy of domains, makes the entire Internet into one namespace. DNS has several top-level domains that are further subdivided into second-level domains. The root of the Internet domain namespace is managed by an Internet authority (currently, the Internet Network Information Center, or InterNIC) that is responsible for delegating administrative responsibility for the top-level domains of the DNS namespace and for registering second-level domain names. The top-level domains are the familiar domain categories commercial (.com), educational (.edu), governmental (.gov), and so forth. Outside the United States, two-letter country-region codes are used, such as .uk for United Kingdom. Second-level domains represent namespaces that are formally registered to institutions (and to individuals) to provide them an Internet presence. Figure 1 shows how a company’s network connects into the Internet DNS namespace.

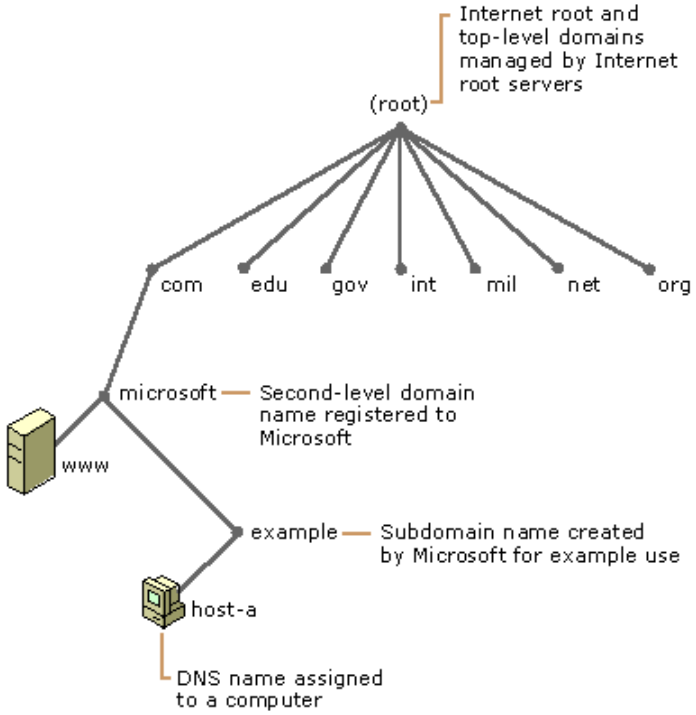

Figure 1. How Microsoft fits into the Internet's DNS namespace

Integration of DNS and Active Directory Namespaces

The integration of DNS and Active Directory is a central feature of the Windows 2000 Server operating system. DNS domains and Active Directory domains use identical domain names for different namespaces. Because the two namespaces share an identical domain structure, it is important to understand that they are *not* the same namespace. Each stores different data and therefore manages different objects. DNS stores its zones[2] and resource records; Active Directory stores its domains and domain objects.

Domain names for DNS are based on the DNS hierarchical naming structure, which is an inverted tree structure: a single root domain, underneath which can be parent and child domains (branches and leaves). For example, a Windows 2000 domain name such as child.parent.microsoft.com identifies a domain named *child,* which is a child domain of the domain named *parent,* itself a child of the domain microsoft.com.

Each computer in a DNS domain is uniquely identified by its fully qualified domain name (FQDN). The FQDN of a computer located in the domain child.parent.microsoft.com is *computername*.child.parent.microsoft.com.

Every Windows 2000 domain has a DNS name (for example,

[2] A DNS zone is a contiguous partition of the DNS namespace that contains the resource records for that zone's DNS domains

OrgName.com), and every Windows 2000-based computer has a DNS name (for example, AcctServer.OrgName.com). Thus, domains and computers are represented both as Active Directory objects and as DNS nodes (a node in the DNS hierarchy represents a domain or a computer).

DNS and Active Directory each uses a database to resolve names:

- **DNS is a name resolution service.** DNS resolves domain names and computer names to IP addresses through requests received by DNS servers as DNS queries to the DNS database. Specifically, DNS clients send DNS name queries to their configured DNS server. The DNS server receives the name query and then either resolves the name query through locally stored files or consults another DNS server for resolution. DNS does *not* require Active Directory to function.
- **Active Directory is a directory service.** Active Directory resolves domain object names to object records through requests received by domain controllers as Lightweight Directory Access Protocol (LDAP)[3] search or modify requests to the Active Directory database. Specifically, Active Directory clients use LDAP to send queries to Active Directory servers. To locate an Active Directory server, an Active Directory client queries DNS. That is, Active Directory uses DNS as a locator service, resolving Active Directory domain, site, and service names to an IP address. For example, to log on to an Active Directory domain, an Active Directory client queries its configured DNS server for the IP address of the LDAP service running on a domain controller for a specified domain. Active Directory *does* require DNS to function.

At the practical level, to understand that the DNS and Active Directory namespaces in a Windows 2000 environment are different is to understand that a DNS host record that represents a specific computer in a DNS zone is in a different namespace than the Active Directory domain computer account object that represents the *same computer*.

In summary, then, Active Directory is integrated with DNS in the following ways:

- **Active Directory domains and DNS domains have the same hierarchical structure.** Although separate and implemented differently for different purposes, an organization's namespace for DNS and Active Directory domains have an identical structure. For example, microsoft.com is a DNS domain and an Active Directory domain.

[3] LDAP is a protocol used to access a directory service; see the sections “LDAP-related Names” and “Lightweight Directory Access Protocol.”

- **DNS zones can be stored in Active Directory.** If you are using the Windows 2000 DNS service, primary zones can be stored in Active Directory for replication to other Active Directory domain controllers and to provide enhanced security for the DNS service.
- **Active Directory clients use DNS to locate domain controllers.** To locate a domain controller for a specified domain, Active Directory clients query their configured DNS server for specific resource records.

Active Directory and the Global DNS Namespace

Active Directory is designed so that it can exist within the scope of the global Internet DNS namespace. When an organization using Windows 2000 Server as its network operating system requires an Internet presence, the Active Directory namespace is maintained as one or more hierarchical Windows 2000 domains beneath a root domain that is registered as a DNS namespace. (An organization can choose not to be part of the global Internet DNS namespace, but if it does so, the DNS service is still required to locate Windows-2000 based computers.)

According to DNS naming conventions, each part of a DNS name that is separated by a period (**.**) represents a node in the DNS hierarchical tree structure and a potential Active Directory domain name in the Windows 2000 domain hierarchical tree structure. As shown in Figure 2, the root of the DNS hierarchy is a node that has a null label (" "). The root of the Active Directory namespace (the forest root) has no parent, and it provides the LDAP entry point to Active Directory.

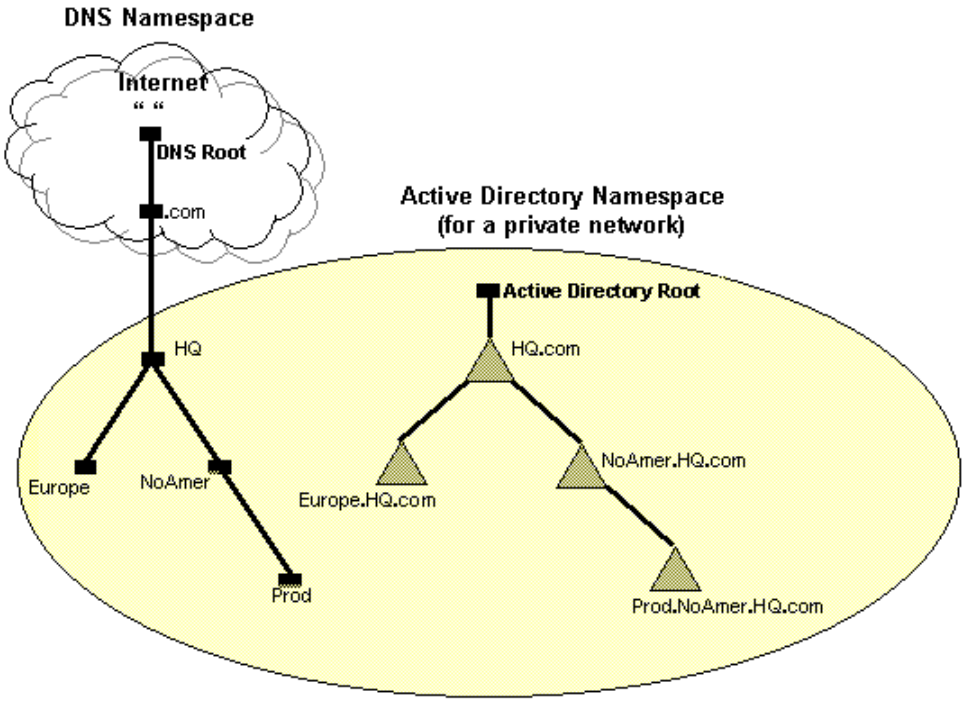

Figure 2. Comparing DNS and Active Directory namespace roots

SRV Resource Records and Dynamic Updates

DNS exists independently of Active Directory, whereas Active Directory is designed specifically to work with DNS. For Active Directory to function properly, DNS servers must support Service Location (SRV) resource records[4]. SRV resource records map the name of a service to the name of a server offering that service. Active Directory clients and domain controllers use SRV resource records to determine the IP addresses of domain controllers.

Note: For more information about planning DNS server deployment in support of your Active Directory domains as well as other deployment issues, see the *Microsoft Windows 2000 Server Deployment Planning Guide* in the "For More Information" section in this paper.

In addition to the requirement that DNS servers in a Windows 2000 network support SRV resource records, Microsoft also recommends that DNS servers provide support for DNS dynamic updates[5]. DNS dynamic updates define a protocol for dynamically updating a DNS server with

[4] Described in the Internet Engineering Task Force (IETF) Internet Draft called draft-ietf-dnsind-rfc2052bis-02.txt, "A DNS RR for specifying the location of services (DNS SRV)". (Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups.)

[5] Described in RFC 2136, Observations on the use of Components of the Class A Address Space within the Internet.

new or changed values. Without the DNS dynamic update protocol, administrators must manually configure the records created by domain controllers and stored by DNS servers.

The new Windows 2000 DNS service supports both SRV resource records and dynamic updates. If you choose to use a non-Windows 2000-based DNS server, you must verify that it supports the SRV resource records or upgrade it to a version that does support them. A legacy DNS server that supports SRV resource records but does not support dynamic updates must have its resource records manually updated at the time you promote a Windows 2000 Server to a domain controller. This is accomplished using the Netlogon.dns file (located in the %systemroot%\System32\config folder), which is created by the Active Directory Installation wizard.

Active Directory Creates Domain Controller

Implementing and administering a network are tangible activities. To understand how Active Directory fits into the picture at the practical level, the first thing you need to know is that installing Active Directory in a computer running the Windows 2000 Server operating system is the act that transforms the server into a domain controller. A domain controller can host exactly one domain.

Specifically, a domain controller is a computer running Windows 2000 Server that has been configured using the Active Directory Installation wizard, which installs and configures components that provide Active Directory directory services to network users and computers. Domain controllers store domain-wide directory data (such as system security policies and user authentication data) and manage user-domain interactions, including user logon processes, authentication, and directory searches.

Promoting a server to a domain controller using the Active Directory Installation wizard also either creates a Windows 2000 domain or adds additional domain controllers to an existing domain.

This section describes what an Active Directory domain controller is and some of the major roles it plays in your network.

With the introduction of Active Directory, Windows 2000 domain controllers function as *peers*. This is a change from the superior/subordinate roles played by Windows NT Server Primary Domain Controllers (PDCs) and Backup Domain Controllers (BDCs). Peer domain controllers support *multimaster replication*, replicating Active Directory information among all domain controllers. The introduction of multimaster replication means that administrators can make updates to Active Directory on any Windows 2000 domain controller in the domain.

In the Windows NT Server operating system, only the PDC has a read-and-write copy of the directory; the PDC replicates a read-only copy of directory information to the BDCs. (For more about multimaster replication, see the section “Multimaster Replication.”)

If you are upgrading to the Windows 2000 operating system from an existing domain, you can perform the upgrade in stages and at your convenience. If you are creating the first domain controller for a new installation, several entities come into being automatically at the same time that Active Directory is loaded. The next two subsections explain the following aspects of installing an Active Directory domain controller in a new network:

- First domain controller is a Global Catalog server.
- First domain controller holds the operations master roles.

Global Catalog

The Windows 2000 operating system introduces the global catalog, a database kept on one or more domain controllers. The global catalog plays major roles in logging on users and querying.

By default, a global catalog is created automatically on the initial domain controller in the Windows 2000 forest, and each forest must have at least one global catalog. If you use multiple sites, you may wish to assign a domain controller in every site to be a global catalog, because a global catalog (which determines an account’s group membership) is required to complete the logon authentication process. This refers to a native-mode domain. Mixed-mode domains do not require a global catalog query for logon.

After additional domain controllers are installed in the forest, you can change the default location of the global catalog to another domain controller using the Active Directory Sites and Services tool. You can optionally configure any domain controller to host a global catalog, based on your organization's requirements for servicing logon requests and search queries. More global catalog servers provide quicker responses to user inquiries; the trade-off is that enabling many domain controllers as global catalog servers increases the replication traffic on the network.

The global catalog performs two key Active Directory roles, logon and querying:

- Logon. In a native-mode domain, the global catalog enables network logon for Active Directory clients by providing universal group membership information[6] for the account sending the logon request

[6] Windows 2000 groups are defined somewhat differently than in Windows NT. Windows 2000 includes two *group types*: 1. Security groups (to manage user and computer access to shared resources and to filter group policy settings); and 2.

to a domain controller. In fact, not just users but every object authenticating to Active Directory must reference the global catalog server, including every computer that boots up. In a multi-domain setup, at least one domain controller that contains the global catalog must be running and available in order for users to log on. A global catalog server must also be available when a user logs on with a non-default user principal name (UPN). (For more about logging on, see the section “Logon Names: UPN and SAM Account Names”).

If a global catalog is not available when a user initiates a network logon process, the user is able to log on only to the local computer (not to the network). The only exception to this is that users who are members of the domain administrators (Domain Admin) group are able to log on to the network even when a global catalog is not available.

- **Querying.** In a forest that contains many domains, the global catalog lets clients quickly and easily perform searches across all domains, without having to search each domain individually. The global catalog makes directory structures within a forest transparent to end-users seeking information. Most Active Directory network traffic is query-related: users, administrators, and programs requesting information about directory objects. Queries occur much more frequently than updates to the directory. Assigning more than one domain controller to be a global catalog server improves response time for users seeking directory information, but you must balance this advantage against the fact that doing so can also increase the replication traffic on your network.

Operations Master Roles

Multimaster replication among peer domain controllers is impractical for some types changes, so only one domain controller, called the operations master, accepts requests for such changes. Because multimaster replication plays an important role in an Active Directory-based network, it is important to know what these exceptions are. In any Active Directory forest, at least five different operations master roles are assigned to the initial domain controller during installation.

Distribution groups (to create e-mail distribution lists). Windows 2000 also includes three *group scopes*: 1. Groups with domain local scope (to define and manage access to resources within a single domain); 2. Groups with global scope (to manage directory objects that require daily maintenance, such as user and computer accounts; you use global scope to group accounts within a domain); and 3. Groups with universal scope (to consolidate groups that span domains; you can add user accounts to groups with global scope and then nest these groups within groups having universal scope). (For more about Windows 2000 groups, including the new universal group type, see the “For More Information” section at the end of this paper.)

When you create the first domain in a new forest, all five of the single master operations roles are automatically assigned to the first domain controller in that domain. In a small Active Directory forest with only one domain and one domain controller, that domain controller continues to own all the operations master roles. In a larger network, whether with one or multiple domains, you can re-assign these roles to one or more of the other domain controllers. Some roles must appear in every forest. Other roles must appear in every domain in the forest.

The following two forest-wide operations master roles must be unique in the forest, that is, there can be only one of each throughout the entire forest:

- **Schema master.** The domain controller holding the schema master role controls all updates and modifications to the schema. The schema defines each object (and its attributes) that can be stored in the directory. To update the schema of a forest, you must have access to the schema master.
- **Domain naming master.** The domain controller holding the domain naming master role controls the addition or removal of domains in the forest.

The following three domain-wide operations master roles must be unique in each domain: there can be only one in each domain in the forest:

- **Relative ID (RID) master.** The RID master allocates sequences of RIDs to each domain controller in its domain. Whenever a domain controller creates a user, group, or computer object, it assigns the object a unique security ID (SID). The security ID consists of a domain security ID (which is the same for all security IDs created in the domain), and a relative ID (which is unique for each security ID created in the domain). When the domain controller has exhausted its pool of RIDs, it requests another pool from the RID Master.
- **Primary domain controller (PDC) emulator.** If the domain contains computers operating without Windows 2000 client software or if it contains Windows NT backup domain controllers (BDCs), the PDC emulator acts as a Windows NT primary domain controller (PDC). It processes password changes from clients and replicates updates to the BDCs. The PDC emulator receives preferential replication of password changes performed by other domain controllers in the domain. If a logon authentication fails at another domain controller due to a bad password, that domain controller forwards the authentication request to the PDC emulator before rejecting the logon attempt.
- **Infrastructure master.** The infrastructure master is responsible for updating all inter-domain references any time an object referenced

by another object moves. For example, whenever the members of groups are renamed or changed, the infrastructure master updates the group-to-user references. When you rename or move a member of a group (and that member resides in a different domain from the group), the group may temporarily appear not to contain that member. The infrastructure master of the group's domain is responsible for updating the group so that it knows the new name or location of the member.

The infrastructure master distributes the update using multimaster replication. Unless there is only one domain controller in the domain, do not assign the infrastructure master role to the domain controller that is hosting the global catalog. If you do, the infrastructure master will not function. If all domain controllers in a domain also host the global catalog (including the situation where only one domain controller exists), all domain controllers have current data and therefore the infrastructure master role is not needed.

Once you have installed an Active Directory domain controller, you have simultaneously also created the initial Windows 2000 domain or added the new domain controller to an existing domain. How do the domain controller and domain fit into the overall network architecture?

This section explains the components of an Active Directory-based network and how they are organized. In addition, it describes how you can delegate administrative responsibility for organizational units (OUs), domains, or sites to appropriate individuals, and how you can assign configuration settings to those same three Active Directory containers. The following topics are covered:

- Objects (including the schema).
- Object naming conventions (including security principal names, SIDs, LDAP-related names, object GUIDs, and logon names).
- Object publishing.
- Domains (including, trees, forests, trusts, and organizational units).
- Sites (including replication).
- How delegation and Group Policy apply to OUs, domains, and sites.

Objects

Active Directory *objects* are the entities that make up a network. An object is a distinct, named set of attributes that represents something concrete, such as a user, a printer, or an application. When you create an Active Directory object, Active Directory generates values for some of the object's attributes, others you provide. For example, when you create a user object, Active Directory assigns the globally unique identifier (GUID), and you provide values for such attributes as the user’s given name,

surname, the logon identifier, and so on.

The Schema

The *schema* is a description of the *object classes* (the various types of objects) and the *attributes* for those object classes. For each class of object, the schema defines the attributes that object class must have, the additional attributes it may have, and the object class that can be its parent. Every Active Directory object is an instance of an object class. Each attribute is defined only once and can be used in multiple classes. For example, the Description attribute is defined once but is used in many different classes.

The schema is stored in Active Directory. Schema definitions are themselves also stored as objects—Class Schema objects and Attribute Schema objects. This lets Active Directory manage class and attribute objects in the same way that it manages other directory objects.

Applications that create or modify Active Directory objects use the schema to determine what attributes the object must or might have, and what those attributes can look like in terms of data structures and syntax constraints.

Objects are either container objects or leaf objects (also called noncontainer objects). A container object stores other objects and a leaf object does not. For example, a folder is a container object for files, which are leaf objects.

Each class of objects in the Active Directory schema has attributes that ensure:

- Unique identification of each object in a directory data store.
- For security principals (users, computers, or groups), compatibility with security identifiers (SIDs) used in the Windows NT 4.0 operating system and earlier.
- Compatibility with LDAP standards for directory object names.

Schema Attributes and Querying

Using the Active Directory Schema tool, you can mark an attribute as indexed. Doing so adds all instances of that attribute to the index, not just the instances that are members of a particular class. Indexing an attribute helps queries find objects that have that attribute more quickly

You can also include attributes in the global catalog. The global catalog contains a default set of attributes for every object in the forest, and you can add your choices to these. Both users and applications use the global catalog to locate objects throughout the forest. Include only those attributes that have the following characteristics:

- **Globally useful.** The attribute should be one that is needed for locating objects (even if just for read access) that may occur

anywhere in the forest.

- **Not volatile.** The attribute should be unchanging or change rarely. Attributes in a global catalog are replicated to all other global catalogs in the forest. If the attribute changes often, significant replication traffic results.
- **Small.** Attributes in a global catalog are replicated to every global catalog in the forest. The smaller the attribute, the lower the impact of that replication.

Schema Object Names

As stated earlier, classes and attributes are both schema objects. Any schema object can be referenced by each of the following types of names:

- **LDAP display name.** The LDAP display name is globally unique for each schema object. The LDAP display name consists of one or more words combined, using initial caps for words after the first word. For example, mailAddress and machinePasswordChangeInterval are the LDAP display names for two schema attributes. Active Directory Schema and other Windows 2000 administrative tools display the LDAP display name of objects, and programmers and administrators use this name to reference the object programmatically. See next subsection for information about programmatically extending the schema; see the section “Lightweight Directory Access Protocol” for more information about LDAP.
- **Common name.** The common name for schema objects is also globally unique. You specify the common name when creating a new object class or attribute in the schema—it is the relative distinguished name (RDN) of the object in the schema that represents the object class. For more about RDNs, see the section “LDAP DN and RDN Names.” For example, the common names of the two attributes mentioned in the preceding paragraph are SMTP-Mail-Address and Machine-Password-Change-Interval.
- **Object identifier (OID).** A schema object’s identifier is a number issued by an issuing authority such as the International Organization for Standardization (ISO) and the American National Standards Institute (ANSI). For example, the OID for the SMTP-Mail-Address attribute is 1.2.840.113556.1.4.786. OIDs are guaranteed to be unique across all networks worldwide. Once you obtain a root OID from an issuing authority, you can use it to allocate additional OIDs. OIDs form a hierarchy. For example, Microsoft has been issued the root OID of 1.2.840.113556. Microsoft manages further branches from this root internally. One of the branches is used to allocate OIDs for Active Directory schema classes, and another for attributes. To continue the example, the OID in Active Directory is

1.2.840.113556.1.5.4, which identifies the Builtin Domain class and can be parsed as shown in Table 1.

Table 1. Object identifier

Object ID Number	Identifies
1	ISO (“root” authority) issued 1.2 to ANSI, then...
2	ANSI issued 1.2.840 to USA, then...
840	USA issued 1.2.840.113556 to Microsoft, then...
113556	Microsoft internally manages several object identifier branches under 1.2.840.113556 that include...
1	a branch called Active Directory that includes...
5	a branch called classes that includes...
4	a branch called Builtin Domain

For more information about OIDs and how to obtain them, see “For More Information” at the end of this document.

Extending the Schema

The Windows 2000 Server operating system provides a default set of object classes and attributes, which are sufficient for many organizations. Although you cannot delete schema objects, you can mark them as deactivated.

Experienced developers and network administrators can dynamically extend the schema by defining new classes and new attributes for existing classes. The recommended way to extend the Active Directory schema is programmatically, through the Active Directory Service Interfaces (ADSI). You can also use the LDAP Data Interchange Format (LDIFDE) utility. (For more about ADSI and LDIFDE, see the sections “Active Directory Service Interface” and “Active Directory and LDIFDE.”)

For development and testing purposes, you can also view and modify the Active Directory schema with the Active Directory Schema tool.

When considering changing the schema, remember these key points:

- Schema changes are global throughout the forest.
- Schema extensions are not reversible (although you can modify some attributes).
- Microsoft requires anyone extending the schema to adhere to the naming rules (discussed in the preceding subsection) for both the LDAP display name and the common name. Compliance is enforced by the Certified for Windows logo program[7].

[7] To qualify for the Certified for Windows logo, your application must be tested by VeriTest for compliance with the Application Specification for Windows 2000. You

- All classes in the schema are derived from the special class Top. With the exception of Top, all classes are subclasses derived from another class. Attribute *inheritance* lets you build new classes from existing classes. The new subclass inherits the attributes of its superclass (parent class).

Extending the schema is an advanced operation. For detailed information about how to extend the schema programmatically, see the section “For More Information” at the end of this document.

Object Naming Conventions

Active Directory supports several formats for object names to accommodate the different forms a name can take, depending on the context in which it is being used (some of the names are in the form of numbers). The following subsections describe these types of naming conventions for Active Directory objects:

- Security principal names.
- Security identifiers (also called security IDs or SIDs).
- LDAP-related names (including DNs, RDNs, URLs, and canonical names).
- Object GUIDs.
- Logon names (including UPN and SAM account names).

If your organization has several domains, it is possible to use the same user name or computer name in different domains. The security ID, GUID, LDAP distinguished name, and canonical name generated by Active Directory uniquely identify each user or computer in the directory. If the user or computer object is renamed or moved to a different domain, the security ID, LDAP relative distinguished name, distinguished name, and canonical name change, but the GUID generated by Active Directory does not change.

---

may choose any combination of platforms, provided that at least one of the Windows 2000 operating systems is included. Applications may carry the "Certified for Microsoft Windows" logo once they have passed compliance testing and have executed a logo license agreement with Microsoft. The logo you receive will indicate the version(s) of Windows for which your product is certified. See the Microsoft Developer Network Web site for more information.

Security Principal Names

A security principal is a Windows 2000 object managed by Active Directory that is automatically assigned a security identifier (SID) for logon authentication and for access to resources. A security principal can be a user account, computer account, or a group, so a security principal name is a name that uniquely identifies a user, computer, or group within a single domain. A security principal object must be authenticated by a domain controller in the domain in which the security principal object is located, and it can be granted or denied access to network resources.

A security principal name is not unique across domains, but, for backward compatibility, it must be unique within its own domain. Security principal objects may be renamed, moved, or contained within a nested domain hierarchy.

The names of security principal objects must conform to the following guidelines:

- The name cannot be identical to any other user, computer, or group name in the domain. It can contain up to 20 uppercase or lowercase characters except for the following: " / \ [ ] : ; | = , + * ? <>
- A user name, computer name, or group name cannot consist solely of periods (.) or spaces.

Security IDs (SIDs)

A security identifier (SID) is a unique number created by the security subsystem of the Windows 2000 operating system, and assigned to security principal objects, that is, to user, group, and computer accounts. Every account on your network is issued a unique SID when that account is first created. Internal processes in the Windows 2000 operating system refer to an account's SID rather than to the account's user or group name.

Each Active Directory object is protected by access control entries (ACEs) that identify which users or groups can access that object. Each ACE contains the SID of each user or group who has permission to access that object and defines what level of access is allowed. For example, a user might have read-only access to certain files, read-and-write access to others, and no access to others.

If you create an account, delete it, and then create an account with the same user name, the new account does not have the rights or permissions previously granted to the old account, because the accounts have different SID numbers.

LDAP-related Names

Active Directory is a Lightweight Directory Access Protocol (LDAP)-compliant directory service. In the Windows 2000 operating system, all

access to Active Directory objects occurs through LDAP. LDAP defines what operations can be performed in order to query and modify information in a directory and how information in a directory can be securely accessed. Therefore, it is LDAP that you use to find or enumerate directory objects and to query or administer Active Directory. (For more about LDAP, see the section “Lightweight Directory Access Protocol.”)

It is possible to query by LDAP distinguished name (which is itself an attribute of the object), but because they are difficult to remember, LDAP also supports querying by other attributes (for example, color to find color printers). This lets you find an object without having to know the distinguished name.

The following three subsections describe Active Directory-supported object-naming formats that are all based on the LDAP distinguished name:

- LDAP DN and RDN names.
- LDAP URLs.
- LDAP-based canonical names.

LDAP DN and RDN Names

LDAP provides *distinguished names* (DNs) and *relative distinguished names* (RDNs) for objects[8]. Active Directory implements these LDAP naming conventions with the variations shown in Table 2.

Table 2. LDAP naming conventions and their Active Directory counterparts

LDAP DN & RDN Naming Convention	Corresponding Active Directory Naming Convention
cn=common name	cn=common name
ou=organizational unit	ou=organizational unit
o=organization	dc=domain component
c=country	(not supported)
Note: cn=, ou=, etc are *attribute types*. The attribute type used to describe an object’s RDN is called the *naming attribute*. The Active Directory naming attributes, shown on the right above, are for the following Active Directory object classes: • cn is used for the *user* object class • ou is used for the *organizational unit* (OU) object class • dc is used for the *domainDns* object class	

[8] Active Directory supports LDAP v2 and LDAP v3, which recognize the RFC 1779 and RFC 2247 naming conventions.

Every Active Directory object has an LDAP DN. Objects are located within Active Directory domains according to a hierarchical *path*, which includes the labels of the Active Directory domain name and each level of container objects. The full path to the object is defined by the DN. The name of the object itself is defined by the RDN. The RDN is that segment of an object's DN that is an attribute of the object itself.

By using the full path to an object, including the object name and all parent objects to the root of the domain, the DN identifies a unique object within the domain hierarchy. Each RDN is stored in the Active Directory database and contains a reference to its parent. During an LDAP operation, the entire DN is constructed by following the references to the root. In a complete LDAP DN, the RDN of the object to be identified appears at the left with the name of the leaf, and it ends at the right with the name of the root, as shown in this example:

cn=JDoe,ou=Widgets,ou=Manufacturing,dc=USRegion,dcOrgName.dc=com

The RDN of the JDoe user object is cn=JDoe, the RDN of Widget (the parent object of JDoe) is ou=Widgets, and so on.

Active Directory tools do not display the LDAP abbreviations for the naming attributes (dc=, ou=, or cn=). These abbreviations are shown only to illustrate how LDAP recognizes the portions of the DN. Most Active Directory tools display object names in canonical form (described later). The Windows 2000 operating system uses the DN to let an LDAP client retrieve an object's information from the directory, but no Windows 2000 user interface requires you to enter DNs. The explicit use of DNs, RDNs, and naming attributes is required only when you are writing LDAP-compliant programs or scripts.

LDAP URL Names

Active Directory supports access using the LDAP protocol from any LDAP-enabled client. RFC 1959 describes a format for an LDAP Uniform Resource Locator (URL) that lets Internet clients have direct access to the LDAP protocol. LDAP URLs are also used in scripting. An LDAP URL begins with the prefix "LDAP," and then it names the server holding Active Directory services followed by the attributed name of the object (the distinguished name). For example:

LDAP://server1.USRegion.OrgName.com/
cn=JDoe,ou=Widgets,ou=Manufacturing,dc=USRegion,dcOrgName,dc=com

LDAP-based Active Directory Canonical Names

By default, Active Directory administrative tools display object names using the *canonical name* format, which lists the RDNs from the root downward and without the RFC 1779 naming attribute descriptors (dc=, ou=, or cn=). The canonical name uses the DNS domain name format,

that is, the constituents of the domain labels section of the name are separated by periods—USRegion.OrgName.com. Table 3 contrasts the LDAP DN with the same name in canonical name format.

Table 3. LDAP DN format contrasted with the canonical name format

Same Name in Two Formats	
LDAP DN Name:	cn=JDoe,ou=Widgets,ou=Manufacturing,dc=USRegion,dcOrgName .dc=com
Canonical Name:	USRegion.OrgName.com/Manufacturing/Widgets/JDoe

Object GUIDs

In addition to its LDAP DN, every object in Active Directory has a *globally unique identifier* (GUID), a 128-bit number assigned by the Directory System Agent when the object is created. The GUID, which cannot be altered or removed, is stored in an attribute, objectGUID, which is a required attribute for every object. Unlike a DN or RDN, which can be changed, the GUID never changes.

When storing a reference to an Active Directory object in an external store (for example, a Microsoft SQL Server™ database), the objectGUID value should be used.

Logon Names: UPN and SAM Account Names

As described earlier, security principals are objects to which Windows-based security is applied for both logon authentication and resource access authorization. Users are one type of security principal. In the Windows 2000 operating system, user security principals require a unique logon name to gain access to a domain and its resources. The next two subsections describe the two types of logon names—UPN and SAM account names.

User Principal Name

In Active Directory, each user account has a *user principal name* (UPN) in the format <*user*>@<*DNS-domain-name*>. A UPN is a friendly name assigned by an administrator that is shorter than the LDAP distinguished name used by the system and easier to remember. The UPN is independent of the user object's DN, so a user object can be moved or renamed without affecting the user logon name. When logging on using a UPN, users no longer have to choose a domain from a list on the logon dialog box.

The UPN's three parts are the UPN prefix (user logon name), the @ character, and the UPN suffix (usually, a domain name). The default UPN suffix for a user account is the DNS name of the Active Directory domain

where the user account is located[9]. For example, the UPN for user John Doe, who has a user account in the OrgName.com domain (if OrgName.com is the only domain in the tree), is JDoe@OrgName.com.The UPN is an attribute (userPrincipalName) of the security principal object. If a user object's userPrincipalName attribute has no value, the user object has a default UPN of userName@DnsDomainName.

If your organization has many domains forming a deep domain tree, organized by department and region, default UPN names can become unwieldy. For example, the default UPN for a user might be sales.westcoast.microsoft.com. The logon name for a user in that domain is *user*@sales.westcoast.microsoft.com. Instead of accepting the default DNS domain name as the UPN suffix, you can simplify both administration and user logon processes by providing a single UPN suffix for all users. (The UPN suffix is used only within the Windows 2000 domain and is not required to be a valid DNS domain name.) You can choose to use your e-mail domain name as the UPN suffix—userName@companyName.com. This gives the user in the example the UPN name of *user*@microsoft.com.

For a UPN–based logon, a global catalog may be necessary, depending on the user logging on, and the domain membership of the user's computer. A global catalog is needed if the user logs on with a non-default UPN and the user's machine account is in a different domain than the user's user account. That is, if, instead of accepting the default DNS domain name as the UPN suffix (as in the example just given, *user*@sales.westcoast.microsoft.com), you provide a single UPN suffix for all users (so that the user then becomes simply *user*@ microsoft.com), a global catalog is required for logon.

You use the Active Directory Domains and Trusts tool to manage UPN suffixes for a domain. UPNs are assigned at the time a user is created. If you have created additional suffixes for the domain, you can select from the list of available suffixes when you create the user or group account. The suffixes appear in the list in the following order:

- Alternate suffixes (if any; last one created appears first).
- Root domain.
- The current domain.

SAM Account Name

A Security Account Manager (SAM) account name is required for compatibility with Windows NT 3.*x* and Windows NT 4.0 domains. The Windows 2000 user interface refers to the SAM account name as the "User logon name (pre-Windows 2000)."

SAM account names are sometimes referred to as flat names because—

[9] If no UPN was added, users can log on by explicitly providing their user name and the DNS name of the root domain.

unlike DNS names—SAM account names do not use hierarchical naming. Because SAM names are flat, each one must be unique in the domain.

Object Publishing

Publishing is the act of creating objects in the directory that either directly contain the information you want to make available or provide a reference to it. For example, a user object contains useful information about users, such as their telephone numbers and e-mail addresses, and a volume object contains a reference to a shared file system volume.

Here are two examples—publishing file and print objects in Active Directory:

- **Share publishing.** You can publish a shared folder as a volume object (also called a shared folder object) in Active Directory, using the Active Directory Users and Groups snap-in. This means that users can now easily and quickly query Active Directory for that shared folder.
- **Printer publishing.** In a Windows 2000 domain, the easiest way to manage, locate, and connect to printers is through Active Directory. By default[10], when you add a printer using the Add Printer wizard and elect to share the printer, Windows 2000 Server publishes it in the domain as an object in Active Directory. Publishing (listing) printers in Active Directory lets users locate the most convenient printer. Users can now easily query Active Directory for any of these printers, searching by printer attributes such as type (PostScript, color, legal-sized paper, and so on) and location. When a printer is removed from the server, it is unpublished by the server.

  You can also publish non-Windows 2000-based printers (that is, printers on non-Windows 2000-based print servers) in Active Directory. To do so, use the Active Directory Users and Computers tool to enter the universal naming convention (UNC) path for the printer. Alternatively, use the Pubprn.vbs script provided in the System32 folder. The Group Policy Downlevel Printer Pruning determines how the pruning service (automatic removal of printers) handles printers on non-Windows 2000-based print servers when a printer is not available.

When to Publish

You should publish information in Active Directory when it is useful or interesting to a large part of the user community and when it needs to be

[10] The group policies that control printer defaults with respect to publishing printers are **Automatically publish new printers in Active Directory** and **Allow printers to be published** (this latter group policy controls whether or not printers on that machine can be published).

highly accessible.

Information published in the Active Directory has two major characteristics:

- **Relatively static.** Publish only information that changes infrequently. Telephone numbers and e-mail addresses are examples of relatively static information suitable for publishing. The user's currently selected e-mail message is an example of highly volatile information.
- **Structured.** Publish information that is structured and can be represented as a set of discrete attributes. A user's business address is an example of structured information suitable for publishing. An audio clip of the user's voice is an example of unstructured information better suited to the file system.

Operational information used by applications is an excellent candidate for publishing in Active Directory. This includes global configuration information that applies to all instances of a given application. For example, a relational database product could store the default configuration for database servers as an object in Active Directory. New installations of that product can then collect the default configuration from the object, simplifying the installation process and enhancing the consistency of installations in an enterprise.

Applications can also publish their connection points in Active Directory. Connection points are used for a client/server rendezvous. Active Directory defines an architecture for integrated service administration using Service Administration Point objects and provides standard connection points for Remote Procedure Call (RPC), Winsock, and Component Object Model (COM)-based applications. Applications that do not use the RPC or Winsock interfaces for publishing their connection points can explicitly publish Service Connection Point objects in Active Directory.

Application data can also be published in the directory using application-specific objects. Application-specific data should meet the criteria discussed above. Data should be globally interesting, relatively non-volatile, and structured.

How to Publish

The means of publishing information varies according to the application or service:

- Remote Procedure Call (RPC). RPC applications use the RpcNs* family of APIs to publish their connection points in the directory and to query for the connection points of services that have published theirs.
- Windows Sockets. Windows Sockets applications use the Registration

and Resolution family of APIs available in Winsock 2.0 to publish their connection points and query for the connection points of services that have published theirs.

- Distributed Component Object Model (DCOM). DCOM services publish their connection points using the DCOM Class Store, which resides in Active Directory. DCOM is the Microsoft Component Object Model (COM) specification that defines how components communicate over Windows-based networks. Use the DCOM Configuration tool to integrate client/server applications across multiple computers. DCOM can also be used to integrate robust Web browser applications.

Domains: Trees, Forests, Trusts, and OUs

Active Directory is made up of one or more domains. Creating the initial domain controller in a network also creates the domain—you cannot have a domain without at least one domain controller. Each domain in the directory is identified by a DNS domain name. You use the Active Directory Domains and Trusts tool to manage domains.

You use domains to accomplish the following network management goals:

- **Delimit security.** A Windows 2000 domain defines a security boundary. Security policies and settings (such as administrative rights and access control lists) do not cross from one domain to another. Active Directory can include one or more domains, each having its own security policies.
- **Replicate information.** A domain is a Windows 2000 directory partition (also called a Naming Context). These directory partitions are the units of replication. Each domain stores only the information about the objects located in that domain. All of a domain's domain controllers can receive changes made to objects, and can replicate those changes to all other domain controllers in that domain.
- **Apply Group Policy.** A domain defines one possible scope for policy (Group Policy settings can also be applied to organizational units or sites). Applying a Group Policy object (GPO) to the domain establishes how domain resources can be configured and used. For example, you can use Group Policy to control desktop settings, such as desktop lockdown and application deployment. These policies are applied only within the domain and not across domains.
- **Structure the network.** Because one Active Directory domain can span multiple sites and can contain millions of objects[11], most organizations do not need to create separate domains to reflect the

[11] Compare this to earlier versions of Windows NT Server, where the SAM database had a limit of about 40,000 objects per domain.

company's divisions and departments. It should never be necessary to create additional domains to handle additional objects. However, some organizations do require more than one domain to accommodate, for example, independent or completely autonomous business units that do not want anyone external to their unit to have authority over their objects. Such organizations can create additional domains and organize them into an Active Directory forest. Another reason to split the network into separate domains is if two parts of your network are separated by a link so slow that you never want complete replication traffic to cross it. (For slow links that can still handle replication traffic on a less frequent schedule, you can configure a single domain with multiple sites.)

- **Delegate administrative authority.** In networks running Windows 2000, you can narrowly delegate administrative authority for individual organizational units as well as for individual domains, which reduces the number of administrators needed with wide administrative authority. Because a domain is a security boundary, administrative permissions for a domain are limited to the domain by default. For example, an administrator with permissions to set security policies in one domain is not automatically granted authority to set security policies in any other domain in the directory.

Understanding domains includes understanding trees, forests, trusts, and organizational units, and how each of these structures relates to domains. Each of these domain components is described in the following subsections:

- Trees
- Forests
- Trust Relationships
- Organizational units

The Windows 2000 operating system also introduces the related concept of sites, but site structure and domain structure are separate—to provide for flexible administration—so sites are handled in a later section. This paper presents the basics about Windows 2000-based domains and sites. For detailed information about how to plan their structure and deployment, see the *Microsoft Windows 2000 Server Deployment Planning Guide* in "For More Information" at the end of this document.

When reading the following subsections describing possible domain structures, keep in mind that for many organizations, a structure consisting of one domain that is simultaneously one forest consisting of one tree is not only possible, but may be the optimal way to organize your network. Always begin with the simplest structure and add complexity only when you can justify doing so.

Trees

In the Windows 2000 operating system, a *tree* is a set of one or more domains with contiguous names. If more than one domain exists, you can combine the multiple domains into hierarchical tree structures. One possible reason to have more than one tree in your forest is if a division of your organization has its own registered DNS name and runs its own DNS servers.

The first domain created is the root domain of the first tree. Additional domains in the same domain tree are child domains. A domain immediately above another domain in the same domain tree is its parent.

All domains that have a common root domain are said to form a *contiguous namespace*. Domains in a contiguous namespace (that is, in a single tree) have contiguous DNS domain names that are formed in the following way: The domain name of the child domain appears at the left, separated from the name of its parent domain to its right by a period. When there are more than two domains, each domain has its parent to its right in the domain name, as shown in Figure 3. Windows 2000-based domains that form a tree are linked by trust relationships that are both two-way and transitive. These trust relationships are described later.

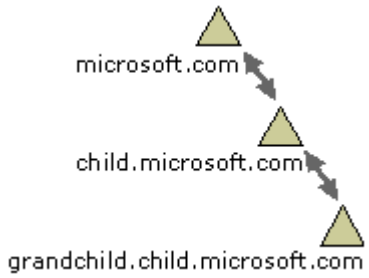

Figure 3. Parent and child domains in a domain tree. Double-headed arrows indicate two-way transitive trust relationships

The parent-child relationship between domains in a domain tree is a naming relationship and a trust relationship only. Administrators in a parent domain are not automatically administrators of a child domain, and policies set in a parent domain do not automatically apply to child domains.

Forests

An Active Directory forest is a *distributed database*, which is a database made up of many partial databases spread across multiple computers. Distributing the database increases network efficiency by letting the data be located where it is most used. The forest's database partitions are defined by domains, that is, a forest consists of one or more domains.

All domain controllers in a forest host a copy of the forest Configuration

and Schema containers in addition to a domain database. A domain database is one part of a forest database. Each domain database contains directory objects, such as security principal objects (users, computers, and groups) to which you can grant or deny access to network resources.

Often, a single forest, which is simple to create and maintain, can meet an organization's needs. With a single forest, users do not need to be aware of directory structure because all users see a single directory through the global catalog. When adding a new domain to the forest, no additional trust configuration is required because all domains in a forest are connected by two-way, transitive trust. In a forest with multiple domains, configuration changes need be applied only once to affect all domains.

You should not create additional forests unless you have a clear need to do so, because each forest you create results in additional management overhead[12]. One possible reason to create more than one forest is if administration of your network is distributed among multiple autonomous divisions that cannot agree on the common management of the schema and configuration containers. Another reason to create a separate forest is to ensure that specific users can never be granted access to certain resources (in a single forest, each user can be included in any group or can appear on a discretionary access control list, or DACL[13], on any computer in the forest). With separate forests, you can define explicit trust relationships to grant users in one forest access to certain resources in the other forest. (For an example of two forests, see Figure 7 in the section "Example: Mixed Environment of Two Forests and One Extranet.")

Multiple domain trees within a single forest do *not* form a contiguous namespace; that is, they have noncontiguous DNS domain names. Although trees in a forest do not share a namespace, a forest does have a single root domain, called the *forest root domain*. The forest root domain is, by definition, the first domain created in the forest. The two forest-wide predefined groups—Enterprise administrators and Schema administrators—reside in this domain.

For example, as shown in Figure 4, although three domain trees (HQ-Root.com, EuropeRoot.com, and AsiaRoot.com) each have a child domain for Accounting named "Acct", the DNS names for these child domains are Acct.HQ-Root.com, Acct.EuropeRoot.com, and Acct.AsiaRoot.com, respectively. There is no shared namespace.

---

[12] For a description of this additional overhead, see the "Microsoft Windows 2000 Server Deployment Planning Guide," which discusses how to plan the structure and deployment of Windows 2000 domains and sites, in the section "For More Information" at the end of this document.

[13] A DACL allows or denies permissions on an object to specific users or groups.

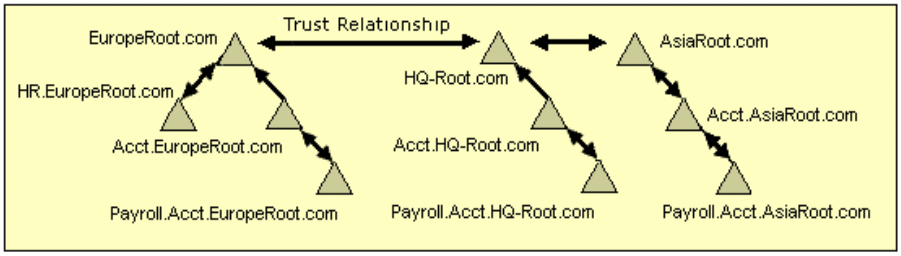

Figure 4. One forest with three domain trees. The three root domains are not contiguous with each other, but EuropeRoot.com and AsiaRoot.com are child domains of HQ-Root.com.

The root domain of each domain tree in the forest establishes a transitive trust relationship (explained in more detail in the next section) with the forest root domain. In Figure 4, HQ-Root.com is the forest root domain. The root domains of the other domain trees, EuropeRoot.com and AsiaRoot.com, have transitive trust relationships with HQ-Root.com. This establishes trust across all the domain trees in the forest.

All Windows 2000 domains in all of the domain trees in a forest possess the following traits:

- Have transitive trust relationships among the domains within each tree.
- Have transitive trust relationships among the domain trees in a forest.
- Share common configuration information.
- Share a common schema.
- Share a common global catalog.

Important: Adding new domains to a forest is easy. However, you cannot move existing Windows 2000 Active Directory domains between forests. You can remove a domain from the forest *only* if it has no child domains. After a tree root domain has been established, you cannot add a domain with a higher-level name to the forest. You cannot create a parent of an existing domain; you can only create a child.

Implementing both domain trees and forests lets you use both contiguous and noncontiguous naming conventions. This flexibility can be useful, for example, in companies with independent divisions that each wants to maintain its own DNS name, such as Microsoft.com and MSNBC.com.

Trust Relationships

A *trust relationship* is a relationship established between two domains that allows users in one domain to be recognized by a domain controller in the other domain. Trusts let users access resources in the other domain and also let administrators administer user rights for users in the other domain. For computers running Windows 2000, account authentication between domains is enabled by two-way, transitive trust

relationships.

All domain trusts in a Windows 2000-based forest are two-way and transitive, defined in the following way:

- **Two-way.** When you create a new child domain, the child domain automatically trusts the parent domain, and vice versa. At the practical level, this means that authentication requests can be passed between the two domains in both directions.
- **Transitive.** A transitive trust reaches beyond the two domains in the initial trust relationship. Here is how it works: If Domain A and Domain B (parent and child) trust each other and if Domain B and Domain C (also parent and child) trust each other, then Domain A and Domain C trust each other (implicitly), even though no direct trust relationship between them exists. At the level of the forest, a trust relationship is created automatically between the forest root domain and the root domain of each domain tree added to the forest, with the result that complete trust exists between all domains in an Active Directory forest. At the practical level, because trust relationships are transitive, a single logon process lets the system authenticate a user (or computer) in any domain in the forest. This single logon process potentially lets the account access resources on any domain in the forest.

Note, however, that the single logon enabled by trusts does not necessarily imply that the authenticated user has rights and permissions in all domains in the forest.

In addition to the forest-wide two-way transitive trusts generated automatically in the Windows 2000 operating system, you can explicitly create the following two additional types of trust relationships:

- **Shortcut Trusts.** Before an account is granted access to resources by a domain controller in another domain, Windows 2000 computes the *trust path* between the domain controllers for the source domain (where the account is located) and the target domain (where the desired resource is located). A trust path is the series of domain trust relationships Windows 2000 security traverses in order to pass authentication requests between any two domains. Computing and traversing a trust path between domain trees in a complex forest can take time. To improve performance, you can explicitly (manually) create a *shortcut trust* between non-adjacent Windows 2000 domains in the same forest. Shortcut trusts are one-way transitive trusts that enable you to shorten the path, as shown in Figure 5. You can combine two one-way trusts to create a two-way trust relationship. Although you cannot revoke the default two-way transitive trusts automatically established among all domains in a Windows 2000

forest, you can delete explicitly created shortcut trusts.

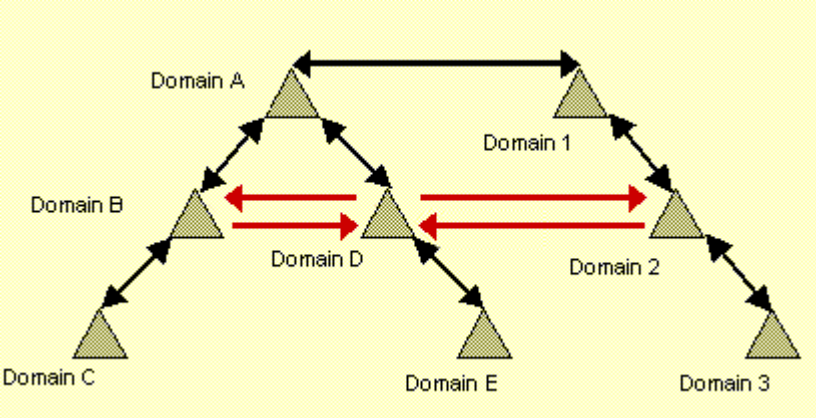

Figure 5. Shortcut trusts between Domains B and D, and between Domains D and 2

- **External Trusts.** External trusts create trust relationships to domains in a different Windows 2000 forest or to a non-Windows 2000 domain (either a Windows NT domain or a Kerberos version 5 realm[14]). External trusts enable user authentication to an external domain. All external trusts are one-way non-transitive trusts, as shown in Figure 6. Again, you can combine two one-way trusts to create a two-way trust relationship.

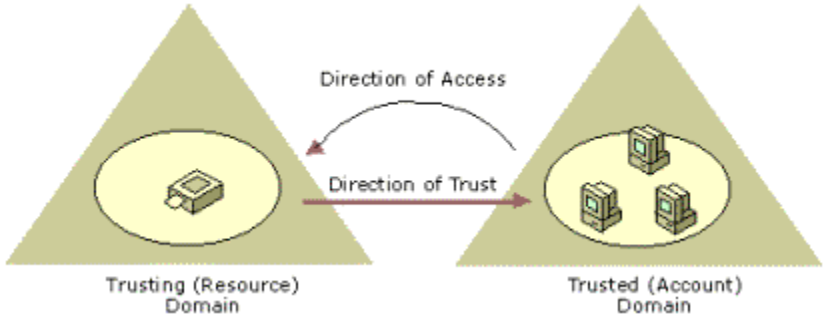

Figure 6. One-way external non-transitive trust

In the Windows NT 4.0 (and earlier) operating system, trust relationships are one-way, and trust is restricted to the two domains between which the trust is established (they are non-transitive). When you upgrade a Windows NT–based domain to a Windows 2000–based one, the existing one-way trust relationships between that domain and any other Windows NT domains are maintained. If you install a new Windows 2000 domain and want to establish trust relationships with Windows NT domains, you

[14] For more about interoperability with Kerberos realms, see the section “Kerberos Role in Interoperability.”

must create Windows 2000 external trusts with those domains. To explicitly establish a trust relationship, you use the Active Directory Domains and Trusts tool.

Example: Mixed Environment of Two Forests and One Extranet

Figure 7 illustrates a mixed environment with two Windows 2000 forests and a Windows NT 4.0 domain. In the figure, four separate namespaces are implemented: A.com, D.com, G.com, and F.

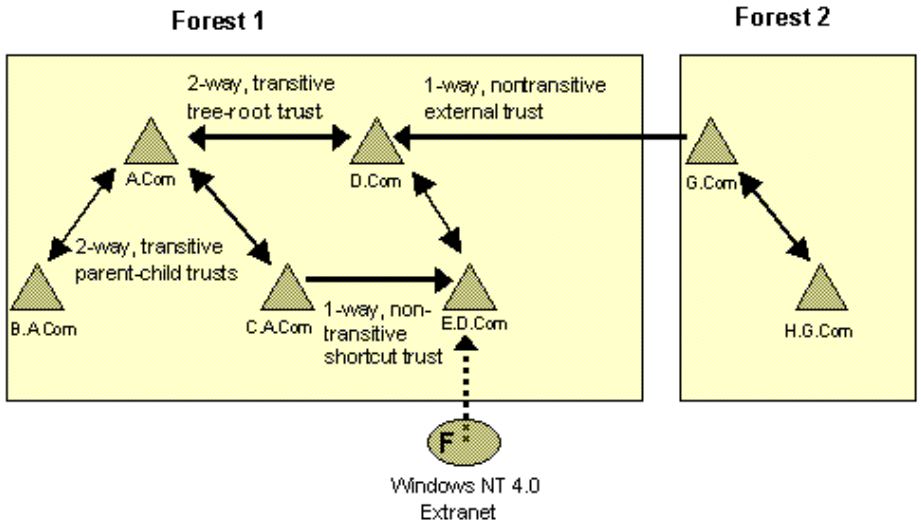

Figure 7. A network with two forests and one extranet

Figure 7 illustrates the following state of affairs:

- A.com and D.com are the roots of separate trees in Forest 1. (A.com is the forest root domain.) The two-way, transitive, tree-root trust between them (automatically generated by Windows 2000) provides complete trust between all domains in the two trees of Forest 1.
- E.D.com frequently uses resources in C.A.com. To shorten the trust path between the two domains, C.A.com trusts E.D.com directly. This one-way, transitive shortcut trust shortens the trust path (reduces the number of hops) for authenticating E.D.com users so they can efficiently use resources in C.A.com.
- G.com is the root of a single tree that makes up Forest 2. The automatic two-way, transitive trust between G.com and H.G.com lets users, computers, and groups in both domains be granted access to each others' resources.
- Domain G.com in Forest 2 implements an explicit one-way external trust relationship with domain D.com in Forest 1 so that users in domain D.com can be granted access to resources in domain G.com. Because the trust is nontransitive, no other domains in Forest 1 can

be granted access to resources in G.com, and users, groups, and computers from D.com cannot be granted access to resources in H.G.com.

- Domain F is a Windows NT 4.0 domain that provides support services to the users in E.D.com. This one-way nontransitive trust does not extend to any other domains in Forest 1. In this scenario, the Windows NT 4.0 domain is an extranet. (An extranet is an intranet that is partly accessible to authorized outsiders. An intranet resides behind a firewall and is inaccessible, but an extranet provides restricted access to people outside the organization.)

Organizational Units

New in the Windows 2000 operating system, organizational units (also called OUs) are a type of directory object into which you can place users, groups, computers, printers, shared folders, and other organizational units within a single domain. An organizational unit (represented as a folder in the Active Directory Users and Computers interface) lets you logically organize and store objects in the domain. If you have multiple domains, each domain can implement its own organizational unit hierarchy.

As Figure 8 illustrates, organizational units can contain other organizational units.

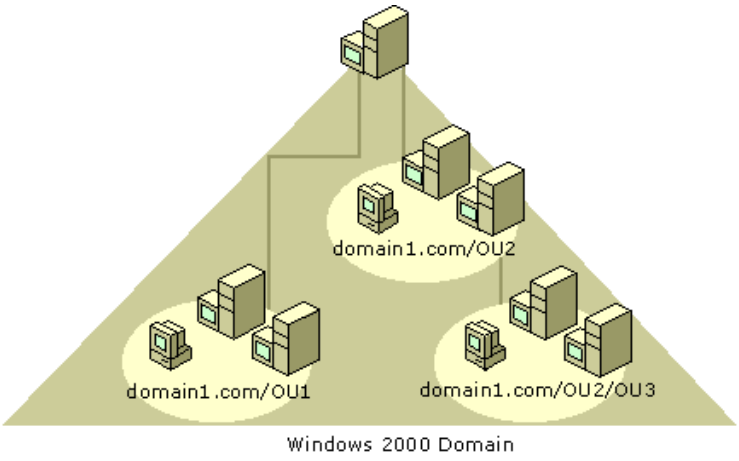

Figure 8. Organizational unit hierarchy inside a single domain

You use organizational units primarily to delegate administrative authority over sets of users, groups, and resources. For example, you might create an organizational unit to contain all user accounts for your entire company. After creating organizational units to delegate administration, apply Group Policy settings to the organizational units to define desktop configurations for users and computers. Because you use organizational units to delegate administration, the structure you create

will probably reflect your administrative model more than your business organization.

Although it is possible for users to navigate a domain's organizational unit structure when looking for resources, querying the global catalog to find resources is much more efficient. Therefore, it is not necessary to create an organizational unit structure that appeals to end-users. It is also possible to create an organizational unit structure that mirrors your business organization, but doing so can prove difficult and expensive to manage. Instead of creating an organizational unit structure to reflect resource location or departmental organization, design organizational units with administrative delegation and Group Policy settings in mind.

For more information about establishing delegation and Group Policy using organizational units, see the section "Use Delegation and Group Policy with OUs, Domains, and Sites." For detailed information about how to design an organizational unit structure when planning how to implement Windows 2000, see the *Microsoft Windows 2000 Server Deployment Planning Guide* in the section "For More Information" at the end of this document.

Sites: Service Clients and Replicate Data

You can think of a Windows 2000-based *site* as a set of computers in one or more IP subnets connected using Local Area Network (LAN) technologies, or as a set of LANs connected by a high-speed backbone. Computers in a single site need to be well-connected, which is generally a characteristic of computers within a subnet. In contrast, separate sites are connected by a link that is slower than LAN speed. You use the Active Directory Sites and Services tool to configure connections both within a site (within a LAN or a set of well-connected LANs) and between sites (in a WAN).

In the Windows 2000 operating system, sites provide the following services:

- Clients can request service from a domain controller in the same site (if one exists).
- Active Directory tries to minimize replication latency for intra-site replication.
- Active Directory tries to minimize bandwidth consumption for inter-site replication.
- Sites let you schedule inter-site replication.

Users and services should be able to access directory information at any time from any computer in the forest. To make this possible, additions, modifications, and deletions of directory data must be relayed (replicated) from the originating domain controller to other domain

controllers in the forest. However, the need to widely distribute directory information must be balanced against the need to optimize network performance. Active Directory sites help maintain this balance.

It is important to understand that sites are independent of domains. Sites map the physical structure of your network, whereas domains (if you use more than one) typically map the logical structure of your organization. Logical and physical structures are independent of each other, which has the following consequences:

- There is no necessary connection between sites and domain namespaces.
- There is no necessary correlation between your network's physical structure and its domain structure. However, in many organizations, domains are set up to reflect physical network structure. This is because domains are partitions, and partitioning influences replication—partitioning the forest into multiple, smaller domains can reduce the amount of replication traffic.
- Active Directory lets multiple domains appear in a single site and a single domain appear in multiple sites.

How Active Directory Uses Site Information

You specify site information using Active Directory Site and Services, and then Active Directory uses this information to determine how best to use available network resources. Using sites makes the following types of operations more efficient:

- **Servicing client requests.** When a client requests a service from a domain controller, it directs the request to a domain controller in the same site, if one is available. Selecting a domain controller that is well connected to the client that placed the request makes handling the request more efficient. For example, when a client logs on using a domain account, the logon mechanism first searches for domain controllers that are in the same site as the client. Attempting to use domain controllers in the client’s site first localizes network traffic, increasing the efficiency of the authentication process.
- **Replicating directory data.** Sites enable the replication of directory data both within and among sites. Active Directory replicates information within a site more frequently than across sites, which means that the best-connected domain controllers, those most likely to need particular directory information, receive replications first. The domain controllers in other sites receive all changes to the directory, but less frequently, reducing network bandwidth consumption. Replicating Active Directory data among domain controllers provides information availability, fault tolerance, load balancing, and performance benefits. (For an explanation of how the

Windows 2000 operating system implements replication, see the subsection “Multimaster Replication” at the end of this section on Sites.)

Domain Controllers, Global Catalogs, and Replicated Data

The information stored in Active Directory on every domain controller (whether or not it is a global catalog server) is partitioned into three categories: domain, schema, and configuration data. Each of these categories is in a separate directory partition, which is also called a Naming Context. These directory partitions are the units of replication. The three directory partitions that each Active Directory server holds are defined as follows:

- **Domain data directory partition.** Contains all of the objects in the directory for this domain. Domain data in each domain is replicated to every domain controller in that domain, but not beyond its domain.
- **Schema data directory partition.** Contains all object types (and their attributes) that can be created in Active Directory. This data is common to all domains in the domain tree or forest. Schema data is replicated to all domain controllers in the forest.
- **Configuration data directory partition.** Contains replication topology and related metadata. Active Directory-aware applications store information in the Configuration directory partition. This data is common to all domains in the domain tree or forest. Configuration data is replicated to all domain controllers in the forest.

If the domain controller is a global catalog server, it also holds a fourth category of information:

- **Partial replica of domain data directory partition for all domains.** In addition to storing and replicating a complete set of all objects in the directory for its own host domain, a global catalog server stores and replicates a partial replica of the domain directory partition for all other domains in the forest. This partial replica, by definition, contains a subset of the properties for all objects in all domains in the forest. (A partial replica is read-only, whereas a complete replica is read/write.)

  If a domain contains a global catalog, other domain controllers replicate all objects in that domain (with a subset of their properties) to the global catalog, and then partial replica replication takes place between global catalogs. If a domain has no global catalog, a regular domain controller serves as the source of the partial replica.

  By default, the partial set of attributes stored in the global catalog includes those attributes most frequently used in search operations, because one of the primary functions of the global catalog is to

support clients querying the directory. Using global catalogs to perform partial domain replication instead of doing full domain replication reduces WAN traffic.

Replication within a Site

If your network consists of a single local area network (LAN) or a set of LANs connected by a high-speed backbone, the entire network can be a single site. The first domain controller you install automatically creates the first site, known as the *Default-First-Site-Name*. After installing the first domain controller, all additional domain controllers are automatically added to the same site as the original domain controller. (Later, if you wish, you can move them to other sites). Here is the only exception: If, at the time you install a domain controller, its IP address falls within the subnet previously specified in an alternative site, the domain controller is then added to this alternative site.

Directory information within a site is replicated frequently and automatically. Intra-site replication is tuned to minimize replication latency, that is, to keep the data as up-to-date as possible. Intra-site directory updates are not compressed. Uncompressed exchanges utilize more network resources but require less domain controller processing power.

Figure 9 illustrates replication within a site. Three domain controllers (one of which is a global catalog) replicate the forest's schema data and configuration data, as well as all directory objects (with a complete set of each object's attributes).

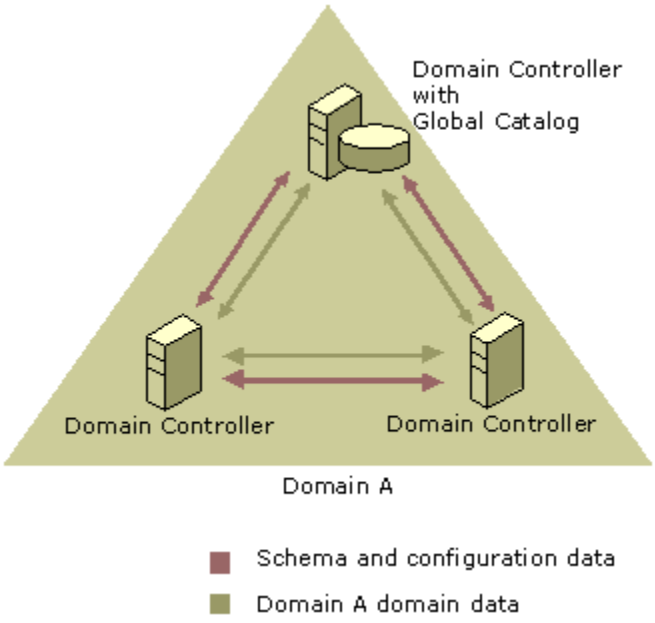

Figure 9. Intra-site replication with just one domain

The configuration formed by the connections used to replicate directory information between domain controllers, called the *replication topology*, is automatically generated by the Knowledge Consistency Checker (KCC) service in Active Directory. Active Directory site topology is a logical representation of a physical network and is defined on a per-forest basis. Active Directory attempts to establish a topology that allows at least two connections to every domain controller, so if a domain controller becomes unavailable, directory information can still reach all online domain controllers through the other connection.

Active Directory automatically evaluates and adjusts the replication topology to meet the changing state of the network. For example, when a domain controller is added to a site, the replication topology is adjusted to incorporate this new addition efficiently.

Active Directory clients and servers use the forest's site topology to route query and replication traffic efficiently.

If you expand your deployment from the first domain controller in one domain to multiple domain controllers in multiple domains (still within one site), the directory information that is replicated changes to include the replication of the partial replica between global catalogs in different domains. Figure 10 shows two domains, each containing three domain controllers. One domain controller in each site is also a global catalog server. Within each domain, the domain controllers replicate the forest's schema data and configuration data, as well as all directory objects (with a complete set of each object's attributes), just as in Figure 9. In addition, each global catalog replicates the directory objects (with only a subset of their attributes) for its own domain to the other global catalog.

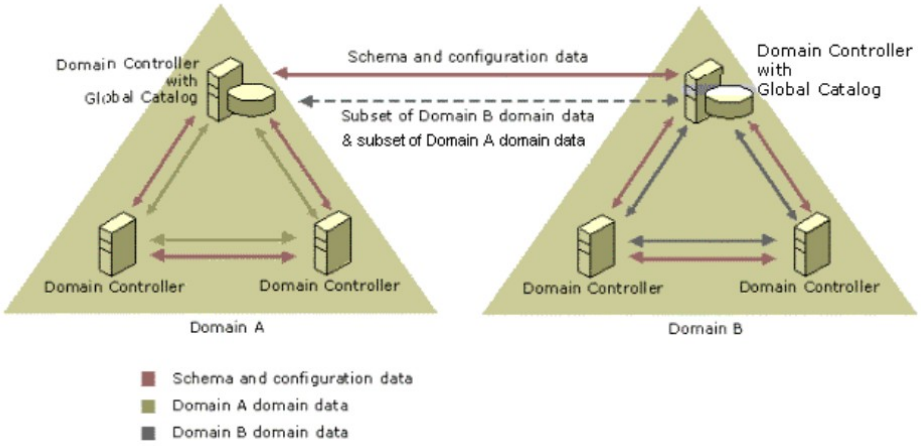

Figure 10. Intra-site replication with two domains and two global catalogs

Create multiple sites to optimize both server-to-server and client-to-server traffic over WAN links. In the Windows 2000 operating system, inter-site replication automatically minimizes bandwidth consumption between sites.

Recommended practices when setting up multiple sites include the following:

- **Geography.** Establish every geographic area that requires fast access to the latest directory information as a site. Establishing areas that require immediate access to up-to-date Active Directory information as separate sites provides the resources required to meet your users' needs.
- **Domain controllers and global catalogs.** Place at least one domain controller in every site, and make at least one domain controller in each site a global catalog. Sites that do not have their own domain controllers and at least one global catalog are dependent on other sites for directory information and are less efficient.

How Sites Are Connected

Network connections between sites are represented by *site links*. A site link is a low-bandwidth or unreliable connection between two or more sites. A WAN that connects two fast networks is an example of a site link. Generally, consider any two networks connected by a link that is slower than LAN speed to be connected by a site link. In addition, a fast link that is near capacity has a low effective bandwidth and is also considered a site link. When you have multiple sites, sites connected by site links become part of the replication topology.

In a Windows 2000-based network, site links are not automatically generated—you must create them using Active Directory Sites and Services. By creating site links and configuring their replication availability, relative cost, and replication frequency, you provide Active Directory with information about what *Connection objects* to create to replicate directory data. Active Directory uses site links as indicators for where it should create Connection objects, and Connection objects use the actual network connections to exchange directory information.

A site link has an associated schedule that indicates at what times of day the link is available to carry replication traffic.

By default, site links are transitive, which means that a domain controller in one site can make replication connections with domain controllers in any other site. That is, if site A is connected to site B, and site B is connected to site C, then domain controllers in site A can communicate with domain controllers in site C. When you create a site, you may want

to create additional links to enable specific connections between sites and customize existing site links connecting the sites.

Figure 11 shows two sites connected by a site link. Of the six domain controllers in the figure, two are bridgehead servers (the bridgehead server role is assigned automatically by the system).

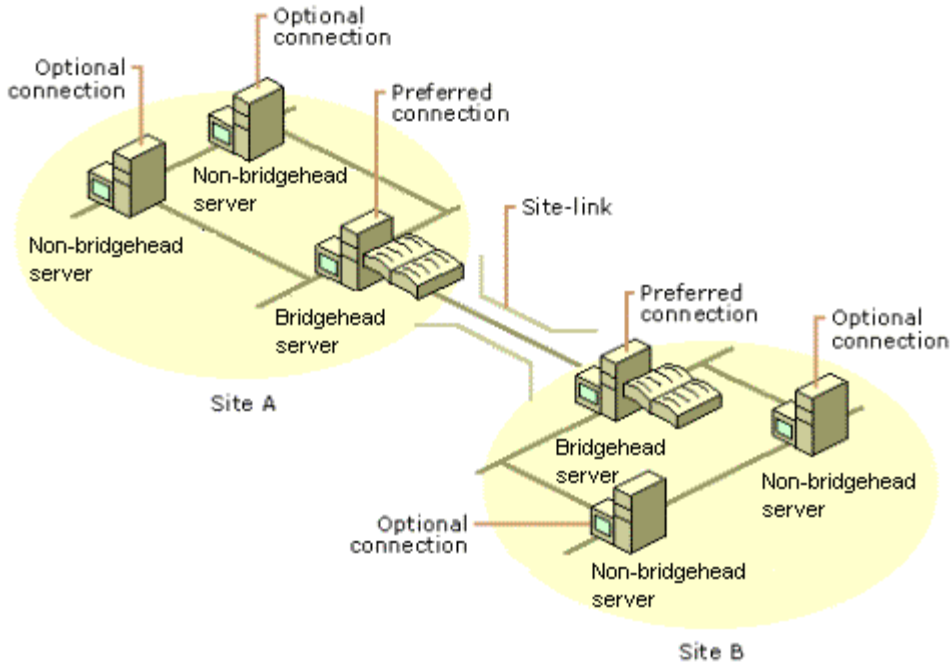

Figure 11. Two sites connected by a site link. Each site's preferred bridgehead server is used preferentially for inter-site information exchange.

The bridgehead servers are the preferred servers for replication, but you can also configure the other domain controllers in the site to replicate directory changes between sites.

After updates are replicated from one site to the bridgehead server in the other site, the updates are then replicated to other domain controllers within the site through intra-site replication. Although a single domain controller receives the initial inter-site directory update, all domain controllers service client requests.

Replication Protocols

Directory information can be exchanged using the following network protocols:

- **IP replication.** IP replication uses remote procedure calls (RPC) for replication within a site (intra-site) and over site links (inter-site). By default, inter-site IP replication adheres to replication schedules. IP replication does not require a certification authority (CA).

- **SMTP replication.** If you have a site that has no physical connection to the rest of you network but that can be reached via Simple Mail Transfer protocol (SMTP), that site has mail-based connectivity only. SMTP replication is used only for replication between sites. You *cannot* use SMTP replication to replicate between domain controllers in the same domain—only inter-domain replication is supported over SMTP (that is, SMTP can be used only for inter-site, inter-domain replication). SMTP replication can be used only for schema, configuration, and global catalog partial replica replication. SMTP replication observes the automatically generated replication schedule.

  If you choose to use SMTP over site links, you must install and configure an enterprise certification authority (CA). The domain controllers obtain certificates from the CA, which the domain controllers then use to sign and encrypt the mail messages that contain directory replication information, ensuring the authenticity of directory updates. SMTP replication uses 56-bit encryption.

Multimaster Replication

Active Directory domain controllers support multimaster replication, synchronizing data on each domain controller, and ensuring consistency of information over time. Multimaster replication replicates Active Directory information among peer domain controllers, each of which has a read-and-write copy of the directory. This is a change from the Windows NT Server operating system, in which only the PDC had a read-and-write copy of the directory (the BDCs received read-only copies from the PDC). Once configured, replication is automatic and transparent.

Update Propagation and Update Sequence Numbers

Some directory services use timestamps to detect and propagate changes. In these systems, it is essential to keep the clocks on all directory servers synchronized. Time synchronization in a network is very difficult. Even with excellent network time synchronization, it is possible for the time at a given directory server to be incorrectly set. This can lead to lost updates.

The Active Directory replication system does not depend on time for update propagation. Instead, it uses Update Sequence Numbers (USNs). A USN is a 64-bit number maintained by each Active Directory domain controller to track updates. When the server writes to any attribute, or property, on an Active Directory object (including the originating write or a replicated write), the USN is advanced and stored with the updated property and with a property that is specific to the domain controller. This operation is performed atomically—that is, the incrementing and storage of the USN and the write of the property value succeed or fail as a single unit.

Each Active Directory-based server also maintains a table of USNs received from replication partners. The highest USN received from each partner is stored in this table. When a given partner notifies the directory server that replication is required, that server requests all changes with USNs greater than the last value received. This simple approach does not depend on the accuracy of timestamps.

Because the USN stored in the table is updated atomically for each update received, recovery after a failure is also simple. To restart replication, a server simply asks its partners for all changes with USNs greater than the last valid entry in the table. Because the table is updated atomically as the changes are applied, an interrupted replication cycle always picks up exactly where it left off, with no loss or duplication of updates.

Collision Detection and Property Version Numbers

In a multimaster replication system such as Active Directory, it is possible for the same property to be updated at two or more different replicas. When a property changes in a second (or third, or fourth, and so on) replica before a change from the first replica has been fully propagated, a replication *collision* occurs. Collisions are detected using property version numbers. Unlike USNs, which are server-specific values, a property version number is specific to the property on an Active Directory object. When a property is first written to an Active Directory object, the property version number is initialized.

Originating writes advance the property version number. An originating write is a write to a property at the system initiating the change. Property writes caused by replication are not originating writes and do not advance the property version number. For example, when a user updates his or her password, an originating write occurs and the password property version number is advanced. Replication writes of the changed password at other servers do not advance the property version number.

A collision is detected when a change is received by replication in which the property version number received is equal to the locally stored version number, and the received and stored values are different. When this occurs, the receiving system applies the update that has the later timestamp. This is the only situation where time is used in replication.

When the received property version number is lower than the locally stored version number, the update is presumed stale and discarded. When the received property version number is higher than the locally stored version number, the update is accepted.

Propagation Dampening

The Active Directory replication system allows loops in the replication topology. This allows the administrator to configure a replication topology

with multiple paths among the servers for performance and availability. The Active Directory replication system performs propagation dampening to prevent changes from propagating endlessly and to eliminate redundant transmission of changes to replicas that are already up-to-date.

The Active Directory replication system employs up-to-date vectors to dampen propagation. The up-to-date vector is a list of server–USN pairs held by each server. The up-to-date vector at each server indicates the highest USN of originating writes received from the server in the server–USN pair. An up-to-date vector for a server in a given site lists all the other servers in that site[15].

When a replication cycle begins, the requesting server sends its up-to-date vector to the sending server. The sending server uses the up-to-date vector to filter changes sent to the requesting server. If the high USN for a given originating writer is greater than or equal to the originating write USN for a particular update, the sending server does not need to send the change; the requesting server is already up-to-date with respect to the originating writer.

Use Delegation and Group Policy with OUs, Domains, and Sites

You can delegate administrative permissions for, and associate Group Policy with, the following Active Directory containers:

- Organizational units
- Domains
- Sites

An organizational unit is the smallest Windows 2000 container to which you can delegate authority or apply Group Policy[16]. Both delegation and Group Policy are security features of the Windows 2000 operating system. This paper briefly discusses them in the limited context of architecture to show that Active Directory structure determines how you use container delegation and Group Policy.

Assigning administrative authority over organizational units, domains, or sites lets you delegate administration of users and resources. Assigning Group Policy Objects (GPOs) to any of these three types of containers lets you set desktop configurations and security policy for the users and computers in the container. The next two subsections discuss these topics in more detail.

---

[15] Up-to-date vectors are not site-specific. An up-to-date vector holds an entry for every server on which the directory partition (Naming Context) is writeable.

[16] In addition to delegating authority over *containers,* you can also grant permissions (such as read/write) down to the attribute level of an *object*.

Container Delegation

In the Windows 2000 operating system, *delegation* allows a higher administrative authority to grant specific administrative rights for organizational units, domains, or sites to groups (or individuals). This greatly reduces the number of administrators needed with sweeping authority over large segments of the user population. Delegating control of a container lets you specify who has permissions to access or modify that object or its child objects. Delegation is one of the most important security features of Active Directory.

Domain and OU Delegation

In the Windows NT 4.0 operating system, administrators sometimes delegate administration by creating multiple domains in order to have distinct sets of domain administrators. In the Windows 2000 operating system, organizational units are easier to create, delete, move, and modify than domains, and they are thus better suited to the delegation role.

To delegate administrative authority (other than authority over sites, covered next), you grant a group specific rights over a domain or organizational unit by modifying the container's discretionary access control list (DACL)[17]. By default, members of the domain administrators (Domain Admin) security group have authority over the entire domain, but you can restrict membership in this group to a limited number of highly trusted administrators. To establish administrators with lesser scope, you can delegate authority down to the lowest level of your organization by creating a tree of organizational units within each domain and delegating authority for parts of the organizational unit subtree.

Domain administrators have full control over every object in their domain. However, they do not have administrative rights over objects in other domains[18].

You delegate administration of a domain or organizational unit by using the Delegation of Control wizard available in the Active Directory Users and Computers snap-in. Right-click the domain or organizational unit, select Delegate Control, add the groups (or users) to whom you want to delegate control, and then either delegate the listed common tasks, or create a custom task to delegate. The common tasks you can delegate are listed in the following table.

---

[17] The access control entries (ACEs) in an object's DACL determine who can access that object and what kind of access they have. When an object is created in the directory, a default DACL (defined in the schema) is applied to it.

[18] By default, the Enterprise Admins group is granted Full Control over all objects in a *forest*.

Domain Common Tasks You Can Delegate	Organizational Unit Common Tasks You Can Delegate
• Join a computer to a domain • Manage Group Policy links	• Create, delete, and manage user accounts • Reset passwords for user accounts • Read all user information • Create, delete, and manage groups • Modify the membership of a group • Manage printers • Create and delete printers • Manage Group Policy links

Using a combination of organizational units, groups, and permissions, you can define the most appropriate administrative scope for a particular group: an entire domain, a subtree of organizational units, or a single organizational unit. For example, you may want to create an organizational unit that lets you grant administrative control for all user and computer accounts in all branches of a single department, such as an Accounting department. Alternatively, you may want to grant administrative control only to some resources within the department, such as computer accounts. A third example is to grant administrative control for the Accounting organizational unit, but not to any organizational units contained within the Accounting organizational unit.

Because organizational units are used for administrative delegation and are not security principals themselves, the parent organizational unit of a user object indicates who manages the user object. It does *not* indicate which resources that particular user can access.

Site Delegation

You use Active Directory Sites and Services to delegate control for sites, server containers, inter-site transports (IP or SMTP), or subnets. Delegating control of one of these entities gives the delegated administrator the ability to manipulate that entity, but it does not give the administrator the ability to manage the users or computers located in it.

For example, when you delegate control of a site, you can choose to delegate control of all objects, or you can delegate control for one or more objects located in that site. The objects for which you can delegate control include User objects, Computer objects, Group objects, Printer objects, Organizational Unit objects, Shared Folder objects, Site objects, Site Link objects, Site Link Bridge objects, and so on. Then, you are

prompted to select the scope of the permissions you want to delegate (general, property-specific, or simply the creation/deletion of specific child objects). If you specify general, you are then are prompted to grant one or more of the following permissions: Full Control, Read, Write, Create All Child objects, Delete All Child objects, Read All Properties, or Write All Properties.

Group Policy

In Windows NT 4.0, you use the System Policy Editor to define user, group, and computer configurations stored in the Windows NT registry database. In the Windows 2000 operating system, Group Policy defines a wider variety of components in the user's environment that administrators can manage. These components include settings for registry-based policies, security options, software deployment options, scripts (for computer startup and shutdown and for user log on and log off), and redirection of special folders[19].

The system applies Group Policy configuration settings to computers at boot time or to users when they log on. Group Policy settings are applied to the users or computers in sites, domains, and organizational units by linking the GPO to the Active Directory container holding the users or computers.

By default, Group Policy affects all users and computers in the linked container. You use membership in security groups to filter which GPOs affect the users and computers in an organizational unit, domain, or site. This lets you apply policy at a more granular level; that is, using security groups lets you apply policy to specific sets of objects within a container. To filter group policy in this way, you use the **Security** tab on a GPO's **Properties** page to control who can read the GPO. Those who do not have Apply Group Policy and Read both set to Allow as members of a security group will not have that GPO applied to them. However, because ordinary users have these permissions by default, Group Policy affects all users and computers in the linked container unless you explicitly change these permissions.

The location of a security group in Active Directory is irrelevant to Group Policy. For the specific container to which the GPO is applied, GPO settings determine the following:

- What domain resources (such as applications) are available to users.
- How these domain resources are configured for use.

For example, a GPO can determine what applications users have available on their computer when they log on, how many users can

[19] You use the Folder Redirection extension to redirect any of the following special folders in a user profile to an alternate location (such as a network share): Application Data, Desktop, My Documents (and/or My Pictures), Start Menu.

connect to Microsoft SQL Server when it starts on a server, or what services users can access when they move to different departments or groups. Group Policy lets you manage a small number of GPOs rather than a large number of users and computers.

Sites, domains, and organizational units, unlike security groups, do not confer membership. Instead, they contain and organize directory objects. Use security groups to grant rights and permissions to users, and then use the three types of Active Directory containers to contain the users and computers and to assign Group Policy settings.

Because resource access is granted using security groups, you might find that using security groups to represent your business organizational structure is more efficient than using domains or organizational units to mirror business structure.

By default, policy settings that are domain-wide or that are applied to an organizational unit containing other organizational units are inherited by the child containers, unless the administrator explicitly specifies that inheritance does not apply to one or more child containers.

Delegating Control of Group Policy

Network administrators (members of the Enterprise Administrators or Domain Administrators group) can use the **Security** tab on the GPO **Properties** page to determine which other administrator groups can modify policy settings in GPOs. To do this, a network administrator first defines groups of administrators (for example, marketing administrators), and then provides them with Read/Write access to selected GPOs. Having full control of a GPO does not enable an administrator to link it to a site, domain, or organizational unit. However, network administrators can also grant that ability using the Delegation of Control wizard.

In the Windows 2000 operating system, you can independently delegate the following three Group Policy tasks:

- Managing Group Policy links for a site, domain, or organizational unit.
- Creating Group Policy objects.
- Editing Group Policy objects.

Group Policy, like most other Windows 2000 administrative tools, is hosted in MMC consoles. The rights to create, configure, and use MMC consoles, therefore, have policy implications. You can control these rights through Group Policy under

`<`*`Group Policy object name`*`>/User Configuration/Administrative Templates/Windows Components/Microsoft Management Console/`

and its subfolders.

Table 4 lists the security permission settings for a Group Policy object.

Table 4. Security Permission Settings for a GPO

Groups (or Users)	Security Permission
Authenticated User	Read with Apply Group Policy ACE
Domain Administrators Enterprise Administrators Creator Owner Local System	Full control without Apply Group Policy ACE

Note: By default, administrators are also authenticated users, which means that they have the Apply Group Policy attribute set.

For detailed information about Group Policy, see the section "For More Information" at the end of this document.

Many companies depend on a diverse collection of technologies that must work together. Active Directory supports a number of standards to ensure interoperability of the Windows 2000 environment with other Microsoft products and with a wide variety of products from other vendors.

This section describes the following types of interoperability supported by Active Directory:

- LDAP protocol.
- Application Programming Interfaces.
- Synchronizing Active Directory with other directory services.
- Virtual and foreign containers' role in interoperability.
- Kerberos role in interoperability.
- Backward compatibility with the Windows NT operating system.

Lightweight Directory Access Protocol

The Lightweight Directory Access Protocol (LDAP) is the industry standard for directory access. LDAP is on the Internet Engineering Task Force (IETF) track for becoming an Internet standard.

Active Directory and LDAP

LDAP is the primary directory access protocol used to add, modify, and delete information stored in Active Directory, as well as to query and retrieve data from Active Directory. The Windows 2000 operating system supports LDAP versions 2 and 3[20]. LDAP defines how a directory client can access a directory server and how the client can perform directory operations and share directory data. That is, Active Directory clients must use LDAP to obtain information from Active Directory or to maintain information in Active Directory.

[20] LDAP version 2 is described in RFC 1777; LDAP version 3 is described in RFC 2251.

Active Directory uses LDAP to enable interoperability with other LDAP-compatible client applications. Given the appropriate permission, you can use any LDAP-compatible client application to browse, query, add, modify, or delete information in Active Directory.

Application Programming Interfaces

You can use the following application programming interfaces (APIs) to access information in Active Directory:

- Active Directory Service Interface (ADSI).
- LDAP C API.

These APIs are described in the next two subsections.

Active Directory Service Interface

Active Directory Service Interface (ADSI) enables access to Active Directory by exposing objects stored in the directory as Component Object Model (COM) objects. A directory object is manipulated using the methods available on one or more COM interfaces. ADSI has a provider-based architecture that allows COM access to different types of directories for which a provider exists.

Currently, Microsoft supplies ADSI providers for Novell NetWare Directory Services (NDS) and NetWare 3, Windows NT, LDAP, and the Internet Information Services (IIS) metabase. (The IIS metabase is the IIS configuration settings.) The LDAP provider can be used with any LDAP directory, including Active Directory, Microsoft Exchange 5.5, or Netscape.

You can use ADSI from many tools, ranging from Microsoft Office applications to C/C++. ADSI supports extensibility so that you can add functionality to an ADSI object to support new properties and methods. For example, you can add a method to the user object that creates an Exchange mailbox for a user when the method is invoked. ADSI has a very simple programming model. It abstracts the data management overhead that is characteristic of non-COM interfaces, such as LDAP C APIs. Because ADSI is fully scriptable, it is easy to develop rich Web applications. ADSI supports ActiveX® Data Objects (ADO) and object linking and embedding database (OLE DB) for querying.

Developers and administrators can add objects and attributes to Active Directory by creating scripts based on ADSI (as well as scripts based on LDIFDE, covered later in this document).

LDAP C API

The LDAP C API, defined in Internet standard RFC 1823, is a set of low-level C-language APIs to the LDAP protocol. Microsoft supports LDAP C APIs on all Windows platforms.

Developers have the choice of writing Active Directory-enabled applications using LDAP C APIs or ADSI. LDAP C APIs are most often used to ease portability of directory-enabled applications to the Windows platform. On the other hand, ADSI is a more powerful language and is more appropriate for developers writing directory-enabled code on the Windows platform.

Synchronizing Active Directory with Other Directory Services

Microsoft provides directory synchronization services that let you synchronize Active Directory information with Microsoft Exchange 5.5, Novell NDS and NetWare, Lotus Notes, and GroupWise. In addition, command-line utilities let you import and export directory information from other directory services.

Active Directory and Microsoft Exchange

The Windows 2000 operating system contains a service called the Active Directory Connector that offers bi-directional synchronization with Microsoft Exchange 5.5. Active Directory Connector provides a rich mapping of objects and attributes when it synchronizes the data between the two directories. For more about Active Directory Connector, see the section “For More Information” at the end of this paper.

Active Directory and Novell NDS and NetWare

As part of Services for Netware 5.0, Microsoft intends to ship a directory synchronization service that performs bi-directional synchronization with Novell NDS and NetWare.

Active Directory and Lotus Notes

As part of Microsoft Exchange 2000 Server, previously code-named “Platinum”, Microsoft intends to ship a directory synchronization service that performs bi-directional synchronization with Lotus Notes for purposes of synchronizing e-mail and other common attributes.

Active Directory and GroupWise

As part of Microsoft Exchange 2000 Server, previously code-named “Platinum”, Microsoft intends to ship a directory synchronization service that performs bi-directional synchronization with GroupWise for purposes of synchronizing e-mail and other common attributes.

Active Directory and LDIFDE

The Windows 2000 operating system provides the command-line utility LDAP Data Interchange Format (LDIFDE) to support importing and exporting of directory information. LDAP Data Interchange Format (LDIF) is an Internet Draft that is an industry standard, which defines the file format used for exchanging directory information. The Windows 2000-based utility that supports import/export to the directory using LDIF is called LDIFDE. LDIFDE lets you export Active Directory information in LDIF

format so that it can later be imported into some other directory. You can also use LDIFDE to import directory information from some other directory.

You can use LDIFDE to perform batch operations, such as add, delete, rename, or modify. You can also populate Active Directory with information obtained from other sources, such as other directory services. In addition, because the schema in Active Directory is stored inside the directory itself, you can use LDIFDE to back up or extend the schema. For a list of LDIFDE parameters and what they do, see Windows 2000 Help. For information about how to use LDIFDE for batch operations with Active Directory, see the section “For More Information” at the end of this document.

Internal and External References

An administrator can create a cross-reference object (cross-ref) that points to a server in a directory external to the forest. When a user searches a subtree that contains this cross-reference object, Active Directory returns a referral to that server as part of the result set, and the LDAP client then chases the referral to get the data requested by the user.

Such references are Active Directory container objects that reference a directory external to the forest. The difference is that an internal reference references an external directory that *does* appear in the Active Directory namespace as a child of an existing Active Directory object, whereas an external reference references an external directory that does *not* appear in the Active Directory namespace as a child.

For both internal and external references, Active Directory contains the DNS name of a server holding a copy of the external directory and the distinguished name of the root of the external directory at which to begin search operations in the external directory.

Kerberos Role in Interoperability

The Windows 2000 operating system supports multiple configurations for cross-platform interoperability:

- **Clients.** A Windows 2000 domain controller can provide authentication for client systems running implementations of RFC-1510 Kerberos, including clients running an operating system other than Windows 2000. Windows 2000-based user and computer accounts can be used as Kerberos principals for Unix-based services.
- **Unix clients and services.** Within a Windows 2000 domain, UNIX clients and servers can have Active Directory accounts and can therefore obtain authentication from a domain controller. In this

scenario, a Kerberos principal is mapped to a Windows 2000 user or computer account.

- **Applications and operating systems.** Client applications for Win32® and operating systems other than Windows 2000 that are based on the General Security Service Application Program Interface (GSS API) can obtain session tickets for services within a Windows 2000 domain.

In an environment that already uses a Kerberos realm, the Windows 2000 operating system supports interoperability with Kerberos services:

- **Kerberos Realm.** Windows 2000 Professional-based systems can authenticate to an RFC-1510 Kerberos server within a realm, with a single sign-on to both the server and a local Windows 2000 Professional account.
- **Trust relationships with Kerberos realms.** A trust relationship can be established between a domain and a Kerberos realm. This means that a client in a Kerberos realm can authenticate to an Active Directory domain to access network resources in that domain.

Backward Compatibility with the Windows NT Operating System

A special type of interoperability is to maintain backward compatibility with earlier versions of the current operating system. The Windows 2000 operating system installs, by default, in a mixed-mode network configuration. A mixed-mode domain is a networked set of computers running both Windows NT and Windows 2000 domain controllers. Because Active Directory supports mixed-mode, you can upgrade domains and computers at whatever rate you choose, based on your organization’s needs.

Active Directory supports the Windows NT LAN Manager (NTLM) authentication protocol used by the Windows NT operating system, which means that authorized Windows NT users and computers can log on to and access resources in a Windows 2000 domain. To Windows NT clients and Windows 95 or 98 clients that are not running Active Directory client software, a Windows 2000 domain appears to be a Windows NT Server 4.0 domain.

Of the many enhancements to the Windows 2000 Server operating system, the introduction of the Active Directory directory service is one of the most significant. Active Directory helps centralize and simplify network manageability and thus improves the network’s ability to support enterprise objectives.

Active Directory stores information about network objects and makes this information available to administrators, users, and applications. It is a

namespace that is integrated with the Internet's Domain Name System (DNS), and, at the same time, it is the software that defines a server as a domain controller.

You use domains, trees, forests, trust relationships, organizational units, and sites to structure the Active Directory network and its objects. You can delegate administrative responsibility for organizational units, domains, or sites to appropriate individuals or groups, and you can assign configuration settings to those same three Active Directory containers. This architecture lets administrators manage the network so that users can concentrate on accomplishing business goals.

Today, it is the norm rather than the exception that companies depend on diverse technologies that need to work together. Active Directory is built on standard directory access protocols, which, together with several APIs, enable Active Directory to interoperate with other directory services and a wide variety of third-party applications. In addition, Active Directory can synchronize data with Microsoft Exchange and provides command-line utilities for importing and exporting data to and from other directory services.

For More Information

For the latest information on the Windows 2000 operating system, check out the Microsoft Windows 2000 Server Web site.In addition, you can look at the following links for more information:

- Windows 2000 Product Help—How to obtain a schema object ID (OID).
- Windows 2000 Platform Software Development Kit—How to use ADSI to extend the schema programmatically.
-
-
-
- Internet Engineering Task Force (IETF) Web site—For IETF RFCs and Internet Drafts.

The *Microsoft Windows 2000 Server Deployment Planning Guide*, which discusses how to plan the structure and deployment of Windows 2000 domains and sites, will be available in bookstores in early 2000. It is also located on the Windows 2000 Server, and Windows 2000 Advanced Server CDs as part of the Support Tools. It is also available on the Windows 2000 web site http://www.microsoft.com/windows2000/library/planning/default.asp

APPENDIX A: TOOLS

This appendix provides a brief overview of the software tools you use to perform the tasks associated with Active Directory.

Microsoft Management Console

In the Windows 2000 Server operating system, Microsoft Management Console (MMC) provides consistent interfaces that let administrators view network functions and use administrative tools. Administrators use the same console whether they are responsible for a single workstation or an entire network of computers. The MMC hosts programs called snap-ins, each of which handles specific network administration tasks. Four of these snap-ins are Active Directory tools.

Active Directory Snap-ins

The Active Directory administrative tools that are included with the Windows 2000 Server operating system simplify directory service administration. You can use the standard tools or use MMC to create custom tools that focus on single management tasks. You can combine several tools into one console. You can also assign custom tools to individual administrators with specific administrative responsibilities.

The following Active Directory snap-ins are available on the Windows 2000 Server Administrative Tools menu of all Windows 2000 domain controllers:

- Active Directory Users and Computers
- Active Directory Domains and Trusts
- Active Directory Sites and Services

The fourth Active Directory snap-in is:

- Active Directory Schema

The recommended way to extend the Active Directory schema is programmatically, through the Active Directory Service Interfaces (ADSI) or the LDAP Data Interchange Format (LDIFDE) utility. However, for development and testing purposes, you can also view and modify the Active Directory schema with the Active Directory Schema snap-in.

Active Directory Schema is not available on the Windows 2000 Server Administrative Tools menu. You must install the Windows 2000 Administration Tools from the Windows 2000 Server compact disc and add it to an MMC console.

A fifth snap-in, which is related to Active Directory tasks, is:

- Group Policy snap-in

Setting group policies is a task related to Active Directory management

of users, computers, and groups. Group Policy objects (GPOs), which contain policy settings, control settings for users and computers in sites, domains, and organizational units. To create or edit GPOs, use the Group Policy snap-in, which is accessed either through Active Directory Users and Computers or through Active Directory Sites and Services (depending on which task you want to perform).

To use the Active Directory administrative tools remotely, from a computer that is not a domain controller (such as one running Windows 2000 Professional), you must install Windows 2000 Administrative Tools.

New Ways to Do Familiar Tasks

Table 5 lists common tasks you can perform using Active Directory snap-ins and related administrative tools. For users of the Windows NT Server operating system, the table also shows where these tasks are performed when using the management tools provided with Windows NT Server 4.0.

Table 5. Tasks performed using Active Directory and Group Policy tools

If you want to:	In Windows NT 4.0, use:	In Windows 2000, use:
Install a domain controller	Windows setup	Active Directory Installation wizard (accessed from Configure Your Server).
Manage user accounts	User Manager	Active Directory Users and Computers
Manage groups	User Manager	Active Directory Users and Computers
Manage computer accounts	Server Manager	Active Directory Users and Computers
Add a computer to a domain	Server Manager	Active Directory Users and Computers
Create or manage trust relationships	User Manager	Active Directory Domains and Trusts.
Manage account policy	User Manager	Active Directory Users and Computers
Manage user rights	User Manager	Active Directory Users and Computers: Edit the Group Policy object for the domain or organizational unit containing the computers to which the user rights apply.
Manage audit policy	User Manager	Active Directory Users and Computers: Edit the Group Policy object assigned to the Domain Controllers organizational unit.
Set policies on users and computers in a site	System Policy Editor	Group Policy, accessed through Active Directory Sites and Services
Set policies on users and computes in a domain	System Policy Editor	Group Policy, accessed through Active Directory Users and Computers
Set policies on users and computers in an organizational unit	Not applicable	Group Policy, accessed through Active Directory Users and Computers
Use Security Groups to filter the scope of policy	Not applicable	Edit the permission entry for **Apply Group Policy** on the security tab of the Group Policy Object's properties sheet.

Active Directory Command-line Tools

Advanced administrators and network support specialists can also use a variety of command-line tools to configure, manage, and troubleshoot Active Directory. These tools are known as the Support Tools and are available on the Windows 2000 Server compact disc in the \SUPPORT\RESKIT folder. They are described in Table 6.

Table 6. Active Directory-related command-line tools

Tool	Description
MoveTree	Move objects from one domain to another.
SIDWalker	Set the access control lists on objects previously owned by accounts that were moved, orphaned, or deleted.
LDP	Allows LDAP operations to be performed against Active Directory. This tool has a graphical user interface.
DNSCMD	Check dynamic registration of DNS resource records, including Secure DNS update, as well as deregistration of resource records.
DSACLS	View or modify the access control lists of directory objects.
NETDOM	Batch management of trusts, joining computers to domains, verifying trusts and secure channels.
NETDIAG	Check end-to-end network and distributed services functions.
NLTest	Check that the locator and secure channel are functioning.
REPAdmin	Check replication consistency between replication partners, monitor replication status, display replication metadata, force replication events and knowledge consistency checker (KCC) recalculation.
REPLMon	Display replication topology, monitor replication status (including group policies), force replication events and knowledge consistency checker recalculation. This tool has a graphical user interface.
DSAStat	Compare directory information on domain controllers and detect differences.
ADSIEdit	A Microsoft Management Console (MMC) snap-in used to view all objects in the directory (including schema and configuration information), modify objects and set access control lists on objects.
SDCheck	Check access control list propagation and replication for specified objects in the directory. This tool enables an administrator to determine if access control lists are being inherited correctly and if access control list changes are being replicated from one domain controller to another.
ACLDiag	Determine whether a user has been granted or denied access to a directory object. It can also be used to reset access control lists to their default state.
DFSCheck	Command-line utility for managing all aspects of Distributed File System (Dfs), checking the configuration concurrency of Dfs servers, and displaying the Dfs topology.

Windows 2000 Command Reference Page

You can find a complete list of Windows 2000 commands, with information about how to use each one, in Windows 2000 Help. Just type “command reference” at either the **Index** tab or the **Search** tab.

Active Directory Service Interface

You can use Active Directory Service Interfaces (ADSI) to create scripts for a wide variety of purposes. The Windows 2000 Server CD contains several sample ADSI scripts. For more about ADSI, see the sections “Active Directory Service Interface” and “For More Information.”